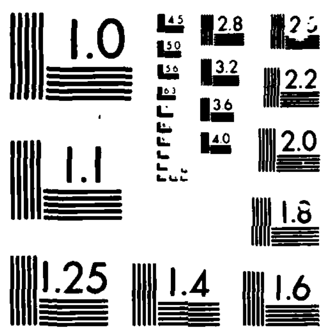
AD-A166 760 WAVE ROTOR RESEARCH: A COMPUTER CODE FOR PRELIMINARY DESIGN OF WAVE DIAGRAMS(U) EXOTECH INC CAMPBELL CA A B MATHUR JUN 85 TR-8502 N00014-84-C-0766 1/1

UNCLASSIFIED F/G 20/4 NL



MICROCOPY CHART

exotech inc.

3rd Floor, 1901 S. Bascom Ave., Campbell,
California, U.S.A. 95008

(12)

AD-A166 760

FINAL REPORT

WAVE ROTOR RESEARCH: A COMPUTER CODE FOR PRELIMINARY DESIGN OF WAVE DIAGRAMS.

TR 8502

JUNE 1985

PREPARED BY

A.B. MATHUR

SUBMITTED TO

NAVAL POSTGRADUATE SCHOOL,
MONTEREY, CALIFORNIA, 93943

UNDER CONTRACT

N00014-84-C-0766

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A166 760
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) Wave Rotor Research: A Computer Code for Preliminary Design of Wave Diagrams.	5. TYPE OF REPORT & PERIOD COVERED Final 1 October 1984-31 May 1985
	6. PERFORMING ORG. REPORT NUMBER TR 8502
7. AUTHOR(s) Atul B. Mathur	8. CONTRACT OR GRANT NUMBER(s) N00014-84-C-0766
9. PERFORMING ORGANIZATION NAME AND ADDRESS Exotech Inc. 1901 S. Bascom Ave., Ste. 337 Campbell, California 95008	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Department of Aeronautics Naval Postgraduate School Monterey, California 93943-5100	12. REPORT DATE June 1985
	13. NUMBER OF PAGES 73
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Office of Naval Research, Code 612A 800 N. Qunicy St. Arlington, Virginia 22217	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

1-D Euler Code
Wave Rotors
Random Choice Method

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A one-dimensional program for solving the unsteady, inviscid, compressible flow in wave rotor devices is described. The Random Choice Method implemented in the code is shown to be very suitable for describing the multiple discontinuities and wave interactions in these flows. The modular structure of the program allows studying different 'families' of wave diagrams quickly and inexpensively. Example applications are included.

DD FORM 1473, 1 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

exotech inc.

3rd Floor, 1901 S. Bascom Ave., Campbell,
California, U.S.A. 95008

FINAL REPORT

WAVE ROTOR RESEARCH: A COMPUTER CODE FOR PRELIMINARY DESIGN OF WAVE DIAGRAMS.

TR 8502

JUNE 1985

PREPARED BY

A.B. MATHUR

SUBMITTED TO

NAVAL POSTGRADUATE SCHOOL,
MONTEREY, CALIFORNIA, 93943

UNDER CONTRACT

N00014-84-C-0766

LIST OF CONTENTS

List of Figures

1. INTRODUCTION

Unsteady flow in the passages of wave rotor devices can adequately be modelled on a one-dimensional basis. However, this modelling can be quite involved due to the peculiar characteristics typical of wave rotor type flows. The numerical calculation has to provide approximate solutions of time-dependent compressible fluid flow problems which involve discontinuities and strong wave interactions. Ref. (1) lists three criteria which such approximate solutions should satisfy simultaneously: (i) the solution must be reasonably accurate in smooth regions of the flow. Continuous waves (rarefaction waves, compression waves) should propagate at the correct speed and should maintain the correct shape which involves steepening or spreading at the correct rate; (ii) discontinuities which are transported along characteristics (gradient discontinuities, contact surfaces), should be modelled by sharp and discrete jumps, and should be transported at the correct speed; and (iii) nonlinear discontinuities such as shocks should be computed stably and accurately.

In addition, the complex pattern of shock waves and contact surfaces that could evolve in wave rotor devices precludes the use of numerical methods which rely on either some type of artificial viscosity or a special treatment of discontinuities. Such methods would quickly become quite impractical for this application due to programming difficulties and cost of execution.

Computation of such solutions has generally been carried out by solving a set of finite difference equations which approximate the governing differential equations of flow. All such schemes inherently have a finite amount of dissipation as well as dispersion of the wave modes they model, and it is difficult to construct difference schemes which simultaneously satisfy the criteria given above. Stability problems may also be an added concern for these schemes.

In view of the foregoing, an alternative approach to solving wave rotor type flows was sought, and the purpose of this report is to describe such a scheme along with some results. The scheme is known variously as Glimm's method, the Random Choice Method (RCM) or the piecewise sampling method. The method evolved from a constructive proof of the existence of solutions to systems of nonlinear hyperbolic conservation laws given by Glimm (Ref. 2). Chorin (Refs. 3 and 4) developed the scheme into an effective numerical tool for gas dynamic applications, with emphasis on detonation combustion problems and reacting gas flows. Although the RCM computes solutions on a fixed grid, it is not a difference scheme, utilizing solutions of locally defined Riemann problems as the basic building blocks for the global solution. Each of the local Riemann problems (defined in more detail in section 2) provides an analytically exact elementary similarity solution. By means of a suitable sampling procedure, usually of a pseudo-random or quasi-random nature, the similarity solutions are superposed to construct the approximate solution to the equations.

With an appropriate sampling technique, the RCM in one dimension is possibly superior to any finite difference scheme in meeting the criteria established above.

2. METHOD

2.1. Solution Procedure

The method models the one-dimensional, compressible, inviscid Euler equations, expressed in conservation form as

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} = 0 \quad , \quad \text{where}$$

$$U(x,t) = \begin{Bmatrix} \rho \\ \rho u \\ E \end{Bmatrix} \quad \text{and} \quad F(U) = \begin{Bmatrix} \rho \\ \rho u^2 + p \\ (E + p)\rho \end{Bmatrix} \tag{1}$$

Here E is the total energy per unit volume and may be expressed as (for a polytropic gas)

$$E = \rho\varepsilon + \frac{1}{2}\rho u^2 \quad , \quad \varepsilon \triangleq \text{internal energy per unit mass}$$
$$= \frac{1}{\gamma-1}\left(\frac{p}{\rho}\right)$$

ρ is the density, p is pressure and u is velocity in the one space dimension being considered here. With initial data specified in the form

$$U(x,0) = \varphi(x) \ ,$$

an initial value problem is defined for the Euler equations. The simplest initial value problem for which discontinuities appear is the Riemann problem: to find the gas flow resulting from an initial state in which the gas on the right of an 'origin' is in a constant state, and the gas on the left is in another constant state, i.e.,

$$\varphi(x) = \begin{matrix} U_L \ , & x < 0 \\ U_R \ , & x > 0 \end{matrix}$$

with

$$U_{L,R} = \begin{Bmatrix} \rho_{L,R} \\ (\rho u)_{L,R} \\ E_{L,R} \end{Bmatrix}$$

where the subsripts L and R denote the left and right sides of the 'origin', here arbitrarily prescibed at 0 . That is, the Riemann problem consists of prescribing constant initial data on either side of an origin where a jump discontinuity exists. As mentioned before, the solution of the problem constitutes a basic building block of the random choice method. A special case of the Riemann problem in which $u_L = u_R = 0$ is often referred to as the shock tube problem. The answer to the problem is that there are four possible types of subsequent flow, depending on the inequalities in the left and right side data prescribed. Thus, in both directions from the origin, a shock or a centered rarefaction wave may propagate, giving rise to the above mentioned four different possibilities. Fig. (1) illustrates the special case of shock tube type flow and the evolution of the wave pattern.

Fig. (2) shows the simple fixed Cartesian grid set up for the method. Let Δx be a spatial increment and Δt a time increment. The solution is to be evaluated at time $(n+1)\Delta t$, n being a non-negative integer, at spatial increments $i\Delta x$, $i = 1,2,3, \ldots$ The initial data is prescribed for each time step at $n\Delta t$ in a piecewise constant manner i.e., it consists of intervals of length Δx where the data is constant, separated by jump discontinuities:

$$U(x\ ,\ n\Delta t) = U_i^n \quad , \ (i-\tfrac{1}{2})\Delta x < x < (i+\tfrac{1}{2})\Delta x$$

The solution at time $(n+1)\Delta t$ then is required to have the same property, i.e., it is piecewise constant over an interval Δx , and it serves as the initial data for the next time step:

$$U(x,(n+1)\Delta t) = U_i^{n+1} \quad , \quad (i - \tfrac{1}{2})\Delta x < x < (i+\tfrac{1}{2})\Delta x$$

This procedure defines a sequence of local Riemann problems to be solved at each time level. On the grid shown in Fig. 2, for example, initial data would be specified at points 1, 3, 5 ...., setting up a succession of Riemann problems defined by each pair of states (1,3) , (3,5) , (5,7), with the discontinuities at the midpoint of each, i.e., at 2, 4, 6, ... etc. If the time step increment Δt is calculated such that

$$\Delta t < \sigma.(\Delta x).\ \max_i\ (|u_i^n|+a_i^n) \quad , \text{ with}$$

$$0 < \sigma < \frac{1}{2}$$

then the waves generated at the discontinuities of adjacent Riemann problems will not interact, as shown schematically in Fig. 2.

Each of the local Riemann problems yields an exact analytical solution, with the resulting wave structure a particular combination/variation of the general structure shown in Fig. 3.

In the x-t plane, the solution to a Riemann problem consists of essentially four regions connected by three waves. Thus states I and IV are the prescribed left and right states for the problem, and states II and III are the 'starred' middle states separated by a slip line or contact discontinuity $\frac{dx}{dt} = u^*$. The velocity, u , and pressure, p , are continuous across the contact, but ρ in general is not. Thus $u_L^* = u_R^*$, $p_L^* = p_R^*$ and $\rho_L^* \neq \rho_R^*$. $S_{1,b}$, $S_{2,b}$ and $S_{1,f}$, $S_{2,f}$ represent respectively the backward and forward facing waves generated at the point of discontinuity and may be either shocks or rarefaction waves.

Still referring to Fig. 3, it is seen that at a time $n\Delta t < t < (n+1)\Delta t$, the exact solution of the local Riemann problem for the interval $[(i-1)\Delta x$, $i\Delta x]$ may actually consist of several distinct states. Consider now a translation of each interval $[(i-1)\Delta x, i\Delta x]$ to $\left[-\frac{\Delta x}{2}, +\frac{\Delta x}{2}\right]$ such that the discontinuity (i.e., the point from which the waves are generated) is centered at a zero origin. Let Θ be the value of a random variable, defined over the interval $[-\frac{1}{2}, +\frac{1}{2}]$, and let

$$\xi = \Theta\Delta x \quad , \quad \text{i.e.} \quad -\frac{\Delta x}{2} < \xi < +\frac{\Delta x}{2}$$

Also, define $U^{n+}_{exact}(x,t)$, $n\Delta t < t < (n+1)\Delta t$, to be the exact solution to each Riemann problem. Using the value of ξ to fix a point in the interval Δx of each Riemann problem, the exact solution at that point is determined and assigned to either the left or the right grid point, depending on whether ξ is $<$ or $>$ 0 . Thus, if the point fixed by ξ is P' (Fig. 3), the exact solution to the Riemann problem at that sampled location is assigned to the grid point on the right and if the sampled point is P", the solution at that location is assigned to the grid point on the left, i.e., for a typical interval $[(i-1)\Delta x, i\Delta x]$,

$$\text{if} \quad \xi < 0 \ , \quad U^{n+1}_{i-1} = U^{n+}_{exact}(\xi, t)$$

$$\text{and if} \quad \xi > 0 \ , \quad U^{n+1}_{i} = U^{n+}_{exact}(\xi, t)$$

It is seen immediately that although the solutions are computed on a grid in this method, it is not a differencing scheme. Also, instead of using a weighted average of the Riemann problem solution to arrive at the solution for a grid point†, the RCM samples a particular value from an explicit wave

† The Godunov method, for example implements

$$U^{n+1}_{i} = \frac{1}{\Delta x}\int_{(i-\frac{1}{2})\Delta x}^{(i+\frac{1}{2})\Delta x} U^{n+}_{exact}(x,t)dx$$

solution, thus eliminating the smoothing out of wave transport and interaction information inherent in averaging. This leads to the 'infinite' resolution of contact discontinuities and shocks that the scheme displays.

From the foregoing discussion, it is evident that the success of the scheme hinges, to a large extent, on the inexpensive and exact solution of Riemann problems and an appropriate sampling technique. Ref. (3) describes a modification to an iterative method due to Godunov (Ref. 5). Theoretical details for the Riemann problem solution are also given in Ref. (6).

The mathematical properties required in a sampling procedure applicable to this scheme are defined in Ref. (1). A brief description of the procedure is given below.

In previous computations using the RCM, random sampling with some variance reduction technique (stratified sampling), was used, i.e., the values were taken from the random number generator installed in the computer (Ref. 3). It was shown in Ref. (1) that a more accurate form of sampling is a technique due to van der Corput (Ref. 7). The sequence generated is, strictly speaking, non-random, but has particular statistical properties that are suitable to the application. The sequence is referred to as quasirandom and is generated as follows:

The binary expansion of natural numbers may be expressed as (with R=2):

$$n = A_0R^0 + A_1R^1 + A_2R^2 + \ldots\ldots + A_mR^m \quad , \quad (0 \leq A_k < R)$$

$$\text{i.e.} \quad n = \sum_{k=0}^{m} A_k . 2^k \quad , \text{ with } A_k = 0 \text{ or } 1 \quad , \quad n = 1, 2, 3, \ldots$$

Next, the digits of the binary numbers are reversed and a decimal point is put preceding the number; this gives the numbers

$$\phi_n = A_0R^{-1} + A_1R^{-2} + \ldots + A_mR^{-(m+1)}$$

$$\text{or,} \quad \phi_n = \sum_{k=0}^{m} A_k . 2^{-(k+1)} \text{ , again with } A_k = 0 \text{ or } 1$$

Conversion to the decimal scale of these numbers yields the required sequence of quasirandom numbers defined over the interval [0,1], i.e.,

$$\phi_n \text{ (decimal)} = \Theta_n + \frac{1}{2}$$

$$\text{or} \quad \Theta_n = \phi_n \text{ (decimal)} - \frac{1}{2}$$

and $\xi_n = \Theta_n . \Delta x$ as defined earlier.

The first few elements of the sequence given below illustrate the construction =

n=1 (decimal) =	1 (binary); ϕ_1 =	0.1 (binary) =	0.5 (decimal)
2	10	0.01	0.25
3	11	0.11	0.75
4	100	0.001	0.125
5	101	0.101	0.625
6	110	0.011	0.375
7	111	0.111	0.875
8	1000	0.0001	0.0625
.	.	.	.
.	.	.	.
.	.	.	.

The van der Corput sequence is 'equidistributed', and yields better results than those obtained using a 'stratified' random sampling technique.

The subroutine employed in the program to compute the random numbers is described in Appendix B.

2.2. Boundary Conditions

In general, the implementation of boundary conditions in the RCM is quite straightforward, but does require some thought. Referring to Fig. 2, the b.c.'s are specified at points 1 and N for the left and right boundary respectively. Note that if the sampled solution at $(n+1)\Delta t$ corresponds to a random number $\xi_n < 0$, the solution is assigned to the grid point on the left. For the Riemann problem defined by points 1 and 3, the sampled solution would then be assigned to grid point 1 at $(n+1)\Delta t$; however this is overridden by assigning the proper boundary condition at 1 again, and there is no contradiction. A similar procedure is adopted at the right hand boundary when $\xi_n > 0$.

The subroutines for the boundary conditions are named in the format BCxn , BC standing for Boundary Condition, x being either L (for Left), or R (for Right boundary) and n being a number from 1 to 5 with the following designations:

1 - solid wall condition

2 - outflow at constant static pressure

3 - special formulation ('piston' inflow)

4 - isentropic inflow from reservoir

5 - special formulation (rarefaction wave cancellation)

2.2.1. Solid Wall Conditions

The solid wall boundary condition requires a zero normal velocity at the wall for inviscid flow computations. Due to the random sampling involved in the method and the lateral movement of the sampled solution $\frac{\Delta x}{2}$ to the left or right of the discontinuity, the condition is difficult to implement uniquely. However, the procedure adopted here is found to yield reasonably accurate results for the applications intended. (Note that the difficulty is not unique to this method only. The implementation of zero mass flux through a surface is difficult per se for the Euler equations).

Referring to Fig. 2, let the physical boundaries be at point 2 and point (N-1) for the left and right sides respectively. However, the boundary conditions are specified at point 1 (point N) for the left (right) side as a fictitious 'mirror' state of the conditions at point 3 (point (n-2)) respectively, but with the reverse sign taken for the velocity component. Thus, for the left hand boundary Riemann problem,

$$p_L = p(3)\ ,\ \rho_L = \rho(3)\ ,\ u_L = -u(3)$$

$$p_R = p(3)\ ,\ \rho_R = \rho(3)\ ,\ u_R = u(3)$$

and, analogously, for the right hand boundary Riemann problem.

$$p_L = p(N-2)\ ,\ \rho_L = \rho(N-2)\ ,\ u_L = u(N-2)$$

$$p_R = p(N-2)\ ,\ \rho_R = \rho(N-2)\ ,\ u_R = -u(N-2)$$

The solutions are then sampled in the manner outlined earlier.

2.2.2. Outflow Conditions

For subsonic outflow, only the static pressure p is defined, with the continuation condition being applied to the rest of the variables. Thus, for the right hand boundary for example, the Riemann problem is defined as follows:

$$p_L = p(N-2)\ ,\ \rho_L = \rho(N-2)\ ,\ u_L = u(N-2)$$

$$p_R = p_{out}\ ,\ \rho_R = \rho(N-2)\ ,\ u_R = -u(N-2)$$

where p_{out} is the specified outlet pressure. If the flow going out is supersonic, there can be no propagation of disturbances upstream, and the continuation condition is implemented for all the variables, i.e., the Riemann problem now is the trivial case defined by

$$p_L = p(N-2)\ ,\ \rho_L = \rho(N-2)\ ,\ u_L = u(N-2)$$

$$p_R = p(N-2)\ ,\ \rho_R = \rho(N-2)\ ,\ u_R = -u(N-2)$$

2.2.3. Special Formulation of 'Piston' Inflow

In general, for idealized wave rotor flows, hot combustion gases are introduced into the rotor through nozzles angled such as to allow the flow to 'slip onto' the rotor, i.e., without incurring incidence or deviation angle losses. Also, in the ideal treatment, the air in the passages of a wave rotor is exposed to the hot gas at high pressure instantaneously. The idealizations allow for uniform conditions to be prescribed at the hot gas inlet port. Thus, a 'special' form of inflow boundary condition needs to be specified here, namely, the static pressure, the velocity and the density of the incoming hot gas. Although equivalent to specifying the total pressure and temperature in the usual inflow boundary condition treatment, some thought is required in wave rotor type flows when specifying p_{gas} , ρ_{gas} and u_{gas} . This is because only a shock wave needs to be generated, with no waves travelling opposite to the direction of flow. The solution to the Riemann problem would then consist of just two states connected by a single shock wave. The flow is equivalent to that generated when a piston is pushed instantaneously into a gas at rest. In general, the state of the air inside the rotor passage is known; explicit relations for two states connected through a shock wave are given in Ref. (6). These so-called transition functions help in specifying the boundary conditions for the incoming flow properly.

If we consider the left boundary for this inflow, the Riemann problem is set up as:

$$p_L = p_{hot\ gas} \ , \ \rho_L = \rho_{hot\ gas} \ , \ u_L = u_{hot\ gas}$$
$$p_R = p(3) \ , \ \rho_R = \rho(3) \ , \ u_R = u(3)$$

with p_L , ρ_L and u_L having been chosen in accordance with the considerations discussed above.

2.2.4. Isentropic Inflow From Reservoir

The induction of fresh charge or air onto the rotor usually corresponds to an isentropic inflow situation. The flow in the vicinity of the passage end can be treated as quasi-steady, with the assumption that no flow separation takes place when the flow enters. Two boundary conditions are required for this type of inflow; these are provided by the conservation of energy in the flow from the external region to the inlet (assumed to be steady), and by the prescibed entropy level of the gas in the external region.

The boundary conditions may thus be expressed as

$$u_{in}^2 + \frac{2}{\gamma-1} a_{in}^2 = \frac{2}{\gamma-1} a_{tot}^2$$

$$S_{in} = S_{tot}$$

where the subscripts 'in' and 'tot' apply to conditions at the inlet of the passage and external reservoir respectively. The sonic velocity is denoted by a , and flow velocity by u . Note that knowledge of the Riemann variable arriving at the passage end from within the passage is required to be able to solve the energy equation above for a_{in} and u_{in} . For the left end, for example,

$$Q_{in} = \frac{2}{\gamma-1} a_{in} - u_{in}$$

which together with the energy equations yields

$$a_{in} = \frac{Q_{in} + \sqrt{\frac{\gamma+1}{\gamma-1} a_{tot}^2 - \frac{\gamma-1}{2} Q_{in}^2}}{\frac{\gamma+1}{\gamma-1}}$$

and subsequently the other variables.

The simple analytical treatment given above has to be modified somewhat if a contact discontinuity is formed when the inflow begins. This is due to the fact that the value of the arriving Riemann variable is changed across such a discontinuity, which thus leads to an additional unknown. Procedures for solving the inflow for these situations are given in Ref. (8). In the program developed here, reasonably good results are obtained by setting the velocity at the boundary point equal to the velocity at the point nearest the physical boundary. For the left end e.g., the variables for the left state of the Riemann problem are obtained as follows:

$$u(1) = u(3) \quad ,$$

a reasonably accurate assumption just at the point of inlet opening.

Then, from the 'energy ellipse',

$$a(1) = \sqrt{a_{tot}^2 - \frac{\gamma-1}{2}\, u(1)^2}$$

$$M(1) = \frac{u(1)}{a(1)} \text{ , incoming Mach number}$$

$$p(1) = \frac{p_{tot}}{\left[1+\frac{\gamma-1}{2} M(1)^2\right]^{\gamma/(\gamma-1)}} \quad ,$$

with similar isentropic relations to compute other flow variables. Note that once the interface or contact discontinuity has moved a certain distance inside the passage, the simple analytical expressions given earlier in the section can be used, since now the value of the arriving Riemann variable would be known at the boundary.

2.2.5. Special Formulation for Rarefaction Wave Cancellation

The spreading of rarefaction fans leads to unwanted wave reflections which occupy large zones in the passages of wave rotors. Fig. (4) shows a wave diagram proposed by Spectra Technology, Inc., which incorporates so-called 'wave management' or 'tuning' ports to ideally cancel (and otherwise attenuate) impinging rarefaction fans. The physical boundary conditions are thus dictated by the flow developing in the passage, i.e., the port has non-uniform flow conditions in it, which at each point match those of the flow at the end of the passage so as to disallow any reflections to take place. Numerically, this condition is achieved by implementing the continuity condition across the boundary for all the flow variables involved. For the left boundary, thus, the Riemann problem is defined by:

$$p_L = p(3) \quad , \quad \rho_L = \rho(3) \quad , \quad u_L = u(3)$$

$$p_R = p(3) \quad , \quad \rho_R = \rho(3) \quad , \quad u_R = u(3)$$

and analogously for the right boundary. Note that these boundary conditions may involve either inflow or outflow.

2.3 Example Calculations

The listing of the program is included in Appendix A, and the various names for the variables are listed in Appendix B, along with some instructions on how to use the program. No effort as yet has been made to optimize the code either for storage requirements or for execution efficiency.

In this section, some sample calculations are carried out using the code, to illustrate its usefulness in constructing idealized design point wave diagrams which can serve as the starting configuration for detailed construction of diagrams incorporating real flow effects.

2.3.1. Test Case for 1-D, Inviscid, Unsteady, Compressible Flow

Fig. (1) illustrates the initial conditions in a shock tube, with the diaphragm at x_0 . Sod (Ref. 9) suggested a test case for hyperbolic conservation laws with the following conditions as initial states in the shock tube:

$$p_1 = 1.0 \quad , \quad \rho_1 = 1.0 \quad , \; u_1 = 0.0$$

$$p_5 = 0.1 \quad , \quad \rho_5 = 0.125 \quad , \quad u_5 = 0.0$$

i.e., the gas on either side of the diaphragm is in a quiescent state initially. The ratio of specific heats is chosen to be 7/5, and Δx is chosen to be 0.01.

The solution (before any wave has reached either the left or right end) is shown in Fig. (5). The squares shown at locations x_1 , x_2 , x_3 and x_4 in the density plot give the analytically calculated amplitude and location of the head - and tail waves of the left-running rarefaction, the contact surface moving to the right and the shock wave moving at supersonic velocity to the right respectively. The solid lines are the solutions obtained by the RCM at different time levels; the zero numerical diffusion feature of the method is evident in the 'infinite' resolution of the contact discontinuity and the shock, and the dispersion (phase error) is within one grid spacing. The constant states are perfectly realized.

It is these features of the method that make it very attractive for application to wave rotor type flows, since the successful design of the device is predicated on being able to accurately compute wave arrival times at the various ports.

2.3.2. Wave Turbine Experiment

Ref. (10) describes the wave rotor experimental set up at the Turbopropulsion Laboratory. Initial tests being carried out currently are with the wave rotor in a turbine mode, i.e., one side of the rotor is blocked off, and high pressure air is brought onto the rotor and taken off again from the other side. The passages of the rotor being angled at 60° to the axis, the 180° reversal in the direction of the fluid flow creates an angular momentum change, in turn generating large turbomachinery work coefficients. Fig. (6) shows the wave diagram computed using the code. The movement of the rotor is from top to bottom. At $t=0$, the high pressure air is brought into contact with quiescent atmospheric air in the rotor passages, at point a. This corresponds to the 'piston' inflow boundary condition described in section 2.2.3.. A shock, S, is generated immediately, (idealized case of instantaneous cell opening), which travels from the right to the left, and strikes the solid wall at the left end. The reflection of the shock takes place at point b according to the solid wall boundary condition described in section 2.2.1.. Behind this shock, and moving at a slower velocity is the contact surface, I, which penetrates into the passage only a fractional distance before encountering the reflected shock, RS, at point c. The reflected shock is transmitted through the contact surface, (bringing the flow to a near halt), and reaches the right side at point d, whereupon the inlet port is closed. The air trapped in the rotor passages is now at a high pressure and in a quiescent state. When this air is released at point e to a low pressure region, a rarefaction wave is generated, R, which travels to the left, spreading out in the process. It interacts with the stationary contact surface, I, setting it into motion again, and reflects off the solid wall at the left as RR. The boundary condition imposed at point e is the outflow at constant static pressure condition described in section 2.2.2.. The outlet port is closed at a time when the exit velocity falls to about half its initial value.

This experiment embodies two fundamental processes in wave rotors: those of cell filling and cell emptying. Almost all the other processes typical to wave rotors are combinations of the cell filling and cell emptying unit processes. Comparison of the ideal computed numbers obtained here with experimental data will provide information helpful in the identification and sources of losses.

The program is set up to start at $t=0$ in this case, with initial data provided along the entire passage, i.e., from $x=0$ to $x=0.1863$m (the actual length of the wave rotor being tested). Since the passages have quiescent atmospheric air in them at $t=0$, the initial data, of course, describes these conditions. Switches for the left and right boundaries describe what type of boundary conditions prevail and direct the program to the appropriate subroutines. These switches, designated SWL and SWR, for left and right respectively, are assigned integer number values which correspond to the numeric value of the particular boundary condition they represent. Thus, if the left boundary is a solid wall, SWL=1, corresponding to the boundary condition subroutine BCL1. In this example then, the initial switch settings at $t=0$ are SWL=1 and SWR=3, corresponding to a solid wall at the left and a 'piston' inflow at the right (which starts at $t=0$ at point a). At point d, the switches are reset to SWL=1 and SWR=1 due to the closure of the inlet port. At point e, the switches are SWL=1, SWR=2, signifying opening of the exhaust port with outflow at a constant static pressure. The whole wave diagram can thus be packaged into a 'module' subroutine and called from the main program with a single call statement. This type of modularity allows for wave diagrams of different 'families' to be developed by simply calling the right 'module' subroutine.

The next two examples illustrate this concept as they deal with two very different types of wave diagrams.

2.3.3. General Electric Wave Engine

Fig. (7) shows a schematic of the wave diagram constructed for the G.E. wave engine. Briefly, the device is configured for a gas generator mode of operation, with counterflow scavenging, and is capable of producing net shaft power. For a fuller description of the machine, see Ref. (11). In this example, fresh charge (air) is induced into the rotor (from an external reservoir) through the wave action of the rarefaction fan originating at the exhaust port opening. The usefulness of the rotor is gauged by the net pressure rise across the machine, i.e., the ratio of the total exhaust pressure to total (fresh air) inlet pressure.

For performance estimation purposes, it is sufficient to investigate only the exhaust and induction processes as shown in Fig. (8). The initial data specified is as follows: the exhausting pressure ratio p_e/p_o , the total pressure ratio across the rotor p_{te}/p_{ta} and an assumed total temperature ratio T_{te}/T_{ta} . In this particular cycle, the amount of fresh charge induced in is ideally equal to the gases exhausted out, i.e., $m_{in} = m_{out}$, and this mass balance is carried out after each computation to correct the assumed temperature ratio T_{te}/T_{ta} (which otherwise constitutes overspecification of the initial conditions).

The calculation starts at $t=0$, with initial data consistent with the chosen pressure and temperature ratios specified along the passage length. Initial switch settings are SWL=1 and SWR=2 for the solid wall boundary at the left and the exhaust to a constant pressure at the right. As shown in the figure, a rarefaction fan is generated, propagating to the left and reflecting off the solid wall. At time $t=\tau_1$, the pressure at the wall has been reduced to that outside the passage, p_{ta} , which is when the inlet port is opened. The switches are now set to SWL=4 and SWR=2 for isentropic inflow from an external reservoir at the left, and still outflow at a constant pressure at the right. The exhaust port is closed at time $t=\tau_2$ which corresponds to the exit velocity having dropped off to approximately half its steady state value at the beginning of the exhaust process. Now the switches are set to SWL=4 and SWR=1 , for the solid wall condition at the right. The sudden closure of the exhaust port generates a 'hammer' shock travelling to the left, interacting with the incoming interface (shown by dashed line), and reaching the passage end at $t=\tau_4$ at which time the inlet port is closed, with the switches being reset to SWL=1 and SWR=1 . Note the reflected shock travelling from left to right generated at the interaction of the contact surface and the hammer shock.

Once this solution is obtained, integration of the mass flux through the inlet and exhaust ports is carried out and if the two numbers do not match, the assumed temperature ratio T_{te}/T_{ta} is adjusted in the initial data, till such time as $m_{in} = m_{out}$.

This calculation is sufficient for performance analyses: if the entire wave diagram has to be worked out, then at a time $t > \tau_3$, hot gas from the combustion chamber is brought onto the rotor (the boundary condition corresponding to 'piston' inflow) on the right hand side. This would generate the shock to compress the induced air and when this shock reached the left end, the transfer port (see Fig. 7) would be opened for such time it takes for the compressed air to be completely scavenged out of the rotor. Fig. (9) shows some performance curves obtained using the procedure outlined above. In Figs. (10a, b, c) are shown three sets of flow parameters at different time steps corresponding to the inlet port just opening, the exhaust port closing and the inlet port closing; the qualitative distributions of the flow parameters in the passage are immediately seen to be accurate when compared with the wave diagram shown in Fig. (8). Of interest is the set of plots for the time step when the inlet port has just been closed. The flow between the end of the passage and the location of the interface is seen to be quite non-uniform in the density plot. At the same time, the shock reflected from the interface has reached the right side and reflected off the solid wall. These considerations help to decide optimum port opening and closing times. For example, Fig. (11) shows what happens if the inlet port is not closed at just the time the shock reaches the end, but rather at some short time later. The shock now sees an open boundary and reflects off as an expansion to match the high pressure behind it with the incoming total pressure which is at a lower value. This reflected expansion is manifested in the pressure, density and velocity plots of the figure.

The entire sequence of wave interactions of this example is computed by the RCM without the implementation of artificial viscosity or artificial compression methods, or tracking and capturing schemes. This 'hands off' feature of the method renders it eminently useful for fast preliminary evaluations of complex wave diagrams for the application at hand.

The next example computes an idealized wave diagram for the nine-port pressure exchanger concept proposed by Spectra Technology, Ref. (12).

2.3.4. Spectra Technology Pressure Exchanger

Fig. (4) shows the ideal wave diagram for the nine-port pressure exchanger. This configuration is a good case example to compute with the RCM because of the different types of boundary conditions that need to be dealt with in the evaluation of the cycle. The computation is started at $t=0$, at the point of high pressure hot gas inlet (driver gas inlet). In the manner described in the G.E. wave engine example, the initial data is prescribed for the entire passage at this time step and the boundary condition switches are initially set at SWL=1 and SWR=3 for the solid wall at the left, and the 'piston' inflow at the right hand end. Since there is a multiplicity of types of boundary conditions, e.g., three outflow ports, an index, JCOUNT, is used to ensure proper sequencing of the switches. The following table is presented as an example of the settings of the switches to carry out calculations for one cycle. The inflow and outflow port conditions are those proposed by Spectra Technology for their idealized diagram.

TIME STEP, N	JCOUNT	SWL	SWR	REMARKS
0	0	1	3	CYCLE STARTS. HP GAS INLET PORT OPENS
500	1	2	3	HP AIR OUTLET PORT OPENS
1408	2	2	1	HP GAS INLET PORT CLOSES
1765	3	5	1	HP AIR OUTLET PORT CLOSES. TUNING PORT L1 OPENS
1816	4	2	1	TUNING PORT L1 CLOSES. IP GAS OUTLET PORT OPENS (PORT E1)
2069	5	2	5	TUNING PORT R1 OPENS
2261	6	2	1	TUNING PORT R1 CLOSES
2595	7	5	1	IP GAS OUTLET PORT CLOSES. TUNING PORT L2 OPENS
2636	8	2	1	TUNING PORT L2 CLOSES. LP GAS OUTLET PORT OPENS (PORT E2)
3029	9	2	5	TUNING PORT R2 OPENS
3237	10	2	4	TUNING PORT R2 CLOSES. LP AIR INLET PORT OPENS
4961	11	1	4	LP GAS OUTLET PORT CLOSES
5529,0	0	1	3	LP AIR INLET PORT CLOSES. CYCLE COMPLETED

The total cycle time as calculated by the RCM is 3.0676 mseconds, which compares well with the time computed by Spectra Technology (using the FCT-SHASTA algorithm) of 3.07 mseconds. The execution time on an IBM 370-3033AP for the 5529 steps computed in the example above was 3 minutes 38 seconds, including the I/O operations and the graphics.

Figs. (12a, b, c) show three sets of plots of the flow parameters for the following cases: a) the H.P. air outlet port opens on time, i.e., just as the shock reaches the left end of the passage, b) the port opens before the shock has reached the end, and c) the port opens after the shock has reached the end. The constant pressure and velocity states that prevail in the passage just after the shock has reached the left end (time 'section' line τ_1 in wave diagram), are perfectly realized in Fig. (12a), while the contact surface is at the location shown by the sharp discontinuities in the density and entropy plots. Should the inlet port be opened earlier, e.g., at the time level shown by τ_{1-} in the wave diagram, what happens is as follows: the pressure in the passage is still at the pre-compressed level and this comes into contact with the pressure level in the port which is considerably higher, resulting in a shock propagating into the passage, colliding with the left moving shock and raising the overall pressure level to ~3.0 as shown in Fig. (12b). However, as soon as the left moving shock reaches the end, it now encounters an open boundary with conditions that do not match those behind the shock, resulting in a rarefaction fan being generated, which propagates to the right. This expansion fan, travelling at sonic velocity relative to the gas into which it is propagating, soon overtakes the right moving shock which is travelling at a subsonic velocity relative to the same gas. This interaction results in an attenuation of both the rarefaction as well as the shock wave. Note that the overall pressure and velocity levels behind the rarefaction are about the same as for case a), i.e., the effects of the mismatch are not very significant at the outlet port. However, should the right moving pressure perturbations of case b) not attenuate each other significantly before they reach the right hand end, the consequences could be severe for the overall wave diagram, since this will lead to further (unwanted) wave reflections.

Fig. (12c) shows what occurs if the outlet port is opened too late, corresponding to time level τ_{l+} on the wave diagram. Now the left travelling shock encounters a wall boundary condition on reaching the left end and reflects off as a shock, effectively doubling the pressure level behind it (>3.5 in pressure plot of Fig. (12c)). When the outlet port opens, there is again a mismatch of conditions in the port and in the passage, with the pressure level in the passage being considerably higher than that prescribed for the outlet port. A rarefaction wave is generated which propagates to the right and overtakes the reflected shock. The same criterion holds for this case too, i.e., the ensuing attenuation of these pressure pulses should occur before they reach the right hand end, preferably even before they reach the interface still propagating towards the left at the flow velocity.

The considerations above give a preview of the nature of decisions required in the successful design of a wave rotor device. It is clear that quite a few iterations are involved in the process of designing a viable wave diagram for a particular application, and each iteration entails calculating two or more complete cycles to ensure 'closure' or repeatability of the cycle. A fast solver like the RCM allows reaching an idealized 'base' design quickly and inexpensively.

Appendix A is a listing of the program in its present development stage. As mentioned earlier, no attempt has been made to optimize the program, either for storage requirements or for execution.

Appendix B gives a description of the structure of the program, a listing of the important variables, the subroutines and the function subprograms. A step by step guide is also included to set up and run the program.

3. DISCUSSION AND RECOMMENDATIONS

3.1. Discussion

For meeting the criteria listed in the Introduction, in one dimension, Glimm's method or the RCM appears to be superior to any difference method. For wave rotor type applications, where discontinuities need to be computed with sharpness, the 'infinite' resolution of such discontinuities inherent in the RCM make it a natural choice to carry out ideal flow calculations for preliminary design purposes. Boundary conditions can be implemented quite easily and do not require information from points outside the domain of dependance as is the case in some finite difference schemes. The van der Corput sampling technique results in the best possible representation of the wave propagation, which is essential for the correct representation of continuous waves, particularly those produced by nonlinear interactions.

The method, however, is not recommended to solve for flows with real effects such as friction, heat transfer and area change, or to be extended to multi-dimensional flows. Although considerable research is being done to rigorously extend the method to such flows, with some degree of success (see Refs. 1, 4, 13), the present state of development is not mature enough to ensure a useful practical code as the outcome.

3.2. Recommendations

Many options are available for one wishing to develop either a 1-D code with real effects and/or a multi-dimensional code for wave rotor type applications. The author prefers to recommend numerical formulations which are dependent on the solution of Riemann problems, such as the Godunov method; the motivating reason for this preference is that a Riemann problem constitutes the solution of a discontinuity in the flow in terms of other discontinuities (if any are, indeed, present), and the scheme is thus intrinsically suited for solving such flows; on the other hand, the other schemes, in general, require to be made aware of discontinuities in the flow through some external device, and then treat them through other artificial devices.

A second-order, quasi one-dimensional (variable cross-sectional area) scheme has recently been developed by Ben-Artzi and Falcovitz (Ref. 14). The method is based on the exact solution of 'generalized Riemann propblems', and has demonstrated very good results; it's least accurate approximation is equivalent to Godunov's first order method (Ref. 9). The resolution of shocks and other disconitinuities and singularities of the flow field is also high. Extension to more than one dimension appears to be straightforward through the use of operator splitting techniques, but has as yet not been tried extensively.

LIST OF REFERENCES

1. Colella, P., "Glimm's Method for Gasdynamics," *SIAM Journal for Scientific Statistical Computations*, Vol. 3, No. 1, March 1982.

2. Glimm, J., "Solutions in the Large for Nonlinear, Hyperbolic Systems of Equations," *Communications in Pure and Applied Mathematics*, No. 18, pp. 167-715, 1965.

3. Chorin, A. J., "Random Choice Solution of Hyperbolic Systems," *Journal of Computational Physics*, No. 22, pp. 517-533, 1976.

4. Chorin, A. J., "Random Choice Methods with Applications to Reacting Gas Flow," *Journal of Computational Physics*, No. 25, pp. 253-272, 1977.

5. Godunov, S. K., "A Finite Difference Method for the Numerical Computation and Discontinuous Solutions of the Equations of Fluid Dynamics," *Mat. Sbornik*, 47, pp. 357-393, 1959.

6. Courant, R. and Friedrichs, K. O., *Supersonic Flow and Shock Waves*, Interscience Publishers Inc., New York, 1948

7. Hammersley, J. M. and Handscomb, D. C., *Monte Carlo Methods*, John Wiley and Sons, Inc., New York, pp. 31-36, 1975.

8. Rudinger, G., *Wave Diagrams for Nonstationary Flow in Ducts*, Van Nostrand, 1955.

9. Sod, G. A., "A Survey of Several Finite Difference Methods for Systems of Non-linear Hyperbolic Conservation Laws," *Journal of Computational Physics*, No. 27, pp. 1-31, 1978.

10. Shreeve, R., Mathur, A., Eidelman, S., and Erwin, J., *Wave Rotor Technology Status and Research Progress Report*, NPS67-82-014PR, Turbopropulsion Laboratory, Naval Postgraduate School, Monterey, California, Novermber 1982.

11. Klapproth, J. F., *Supercharged Turbowave Engines*, General Electric R62FPD171, General Electric Flight Propulsion Division, Everdale, Ohio, May 1962.

12. Taussig, R., "Wave Rotor Turbofan Engines for Aircraft," *Mechanical Engineering*, Vol. 106, No. 11, pp. 60-68, November 1984.

13. Gottlieb, J. J. and Igra, O., "Interaction of Rarefaction Waves with Area Reductions in Ducts," *Journal of Fluid Mechanics*, Vol. 137, pp. 285-305, 1983.

14. Ben Artzi, M. and Falcovitz, J., "A Second-Order Godunov Type Scheme for Compressible Fluid Dynamics," *Journal of Computational Physics*, No. 55, pp. 1-32, 1984.

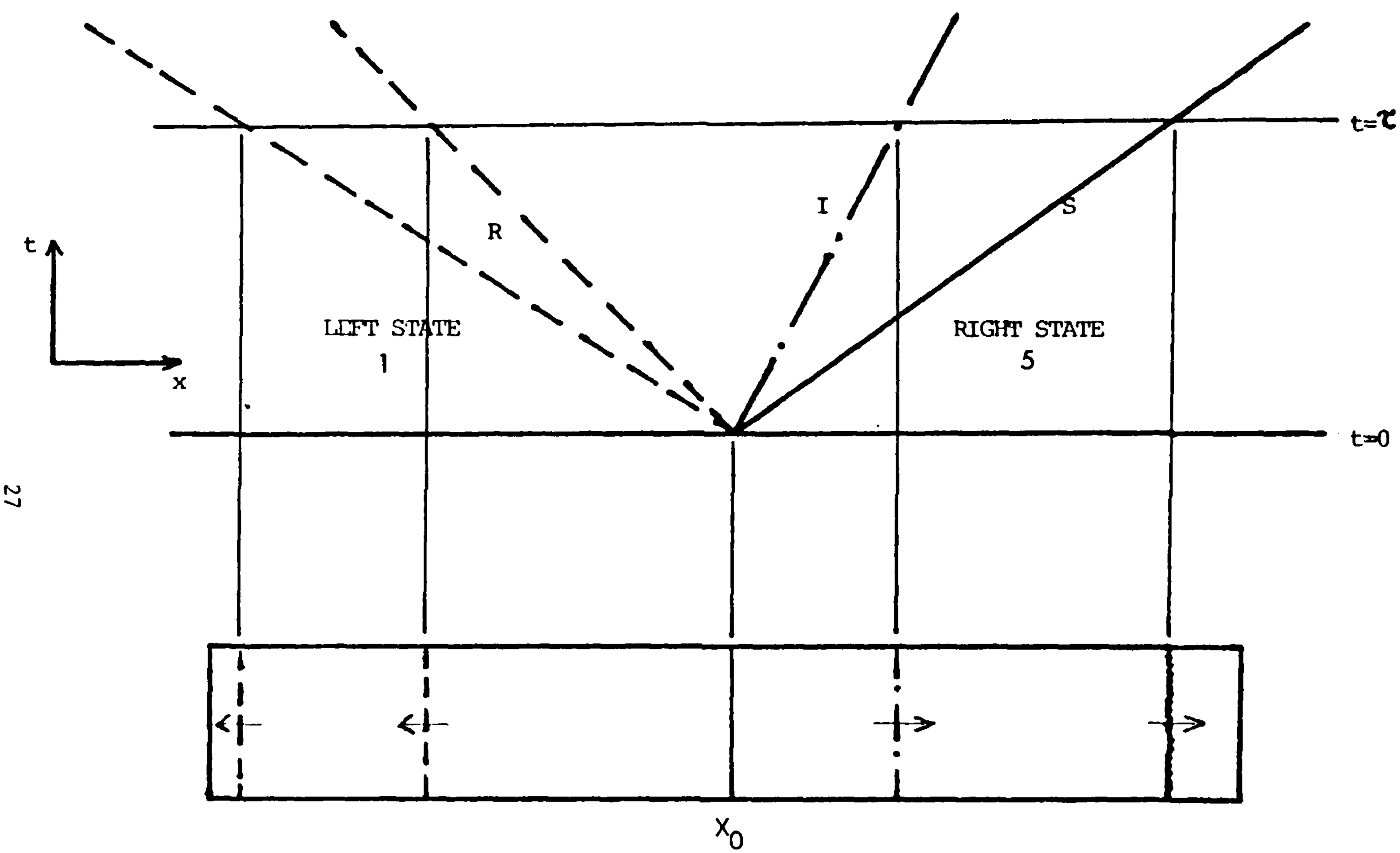


FIG.1: THE RIEMANN PROBLEM IN GAS DYNAMICS
SPECIAL CASE OF SHOCK-TUBE FLOW

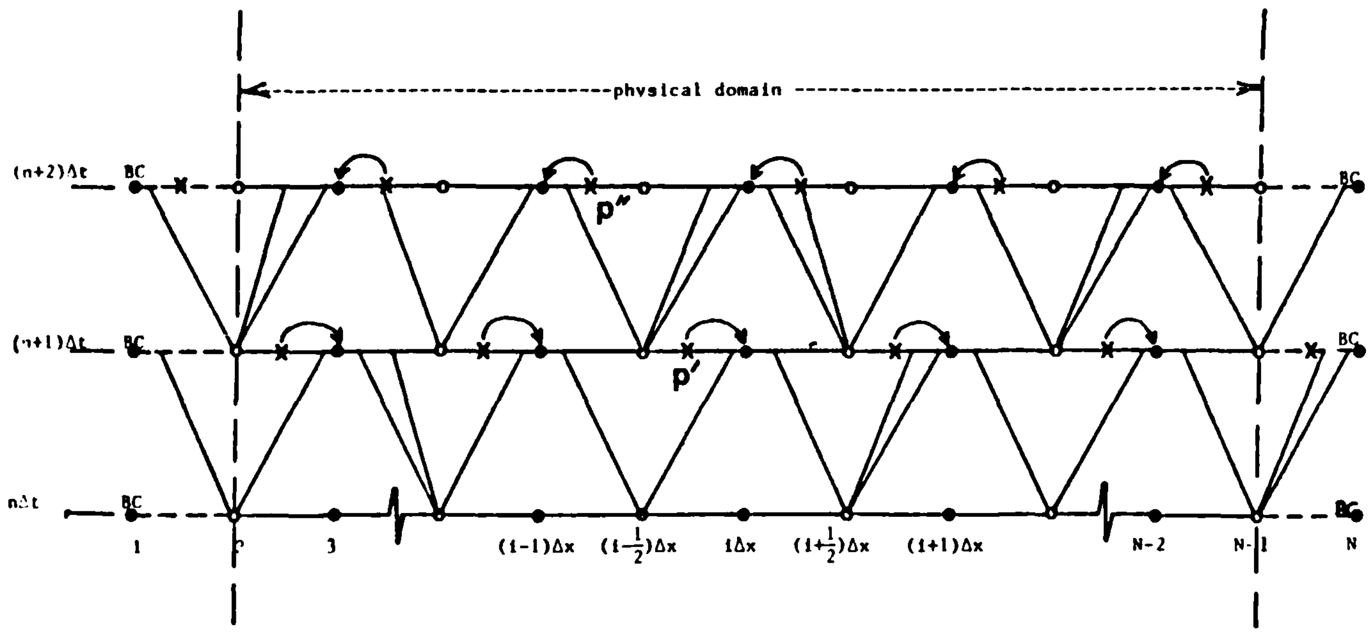


Fig.2 : Cartesian Grid for 1-D Random Choice Method.
●- solution grid points
o- intermediate grid points
BC- points where boundary conditions are specified
X- location where local Riemann Problem solution is sampled

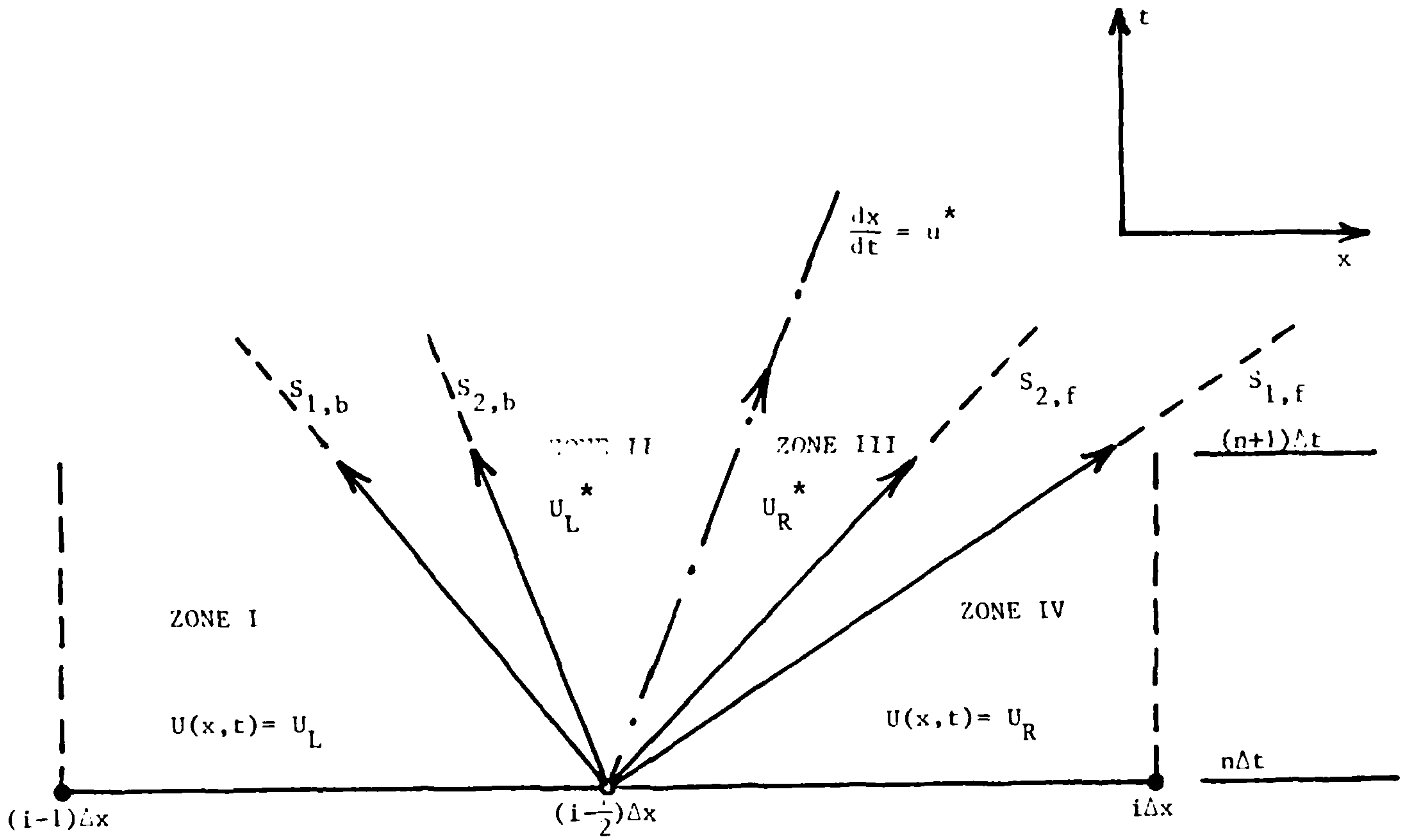


Fig.3 : General Wave Structure Resulting From the Solution of a Riemann Problem

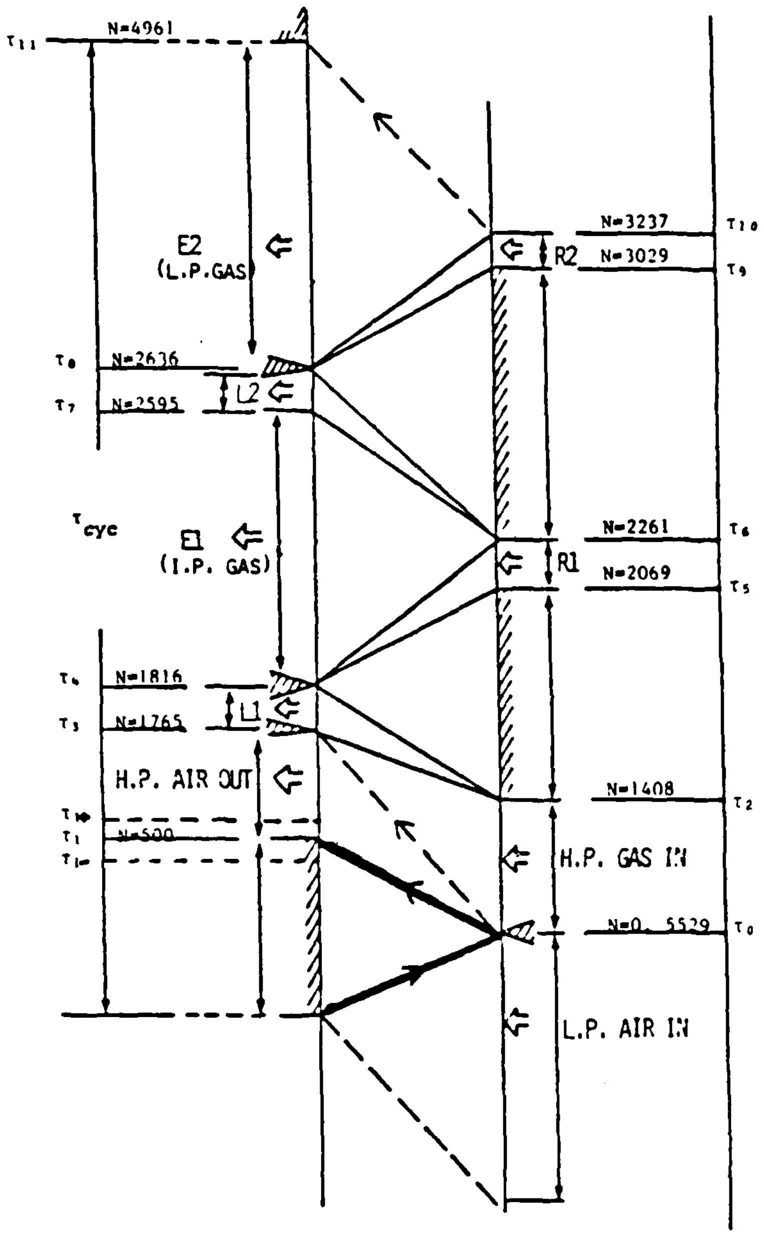


Fig.4 : Ideal Wave Diagram for Pressure Exchanger
(Spectra Technology).
LP,IP,HP: Low, Intermediate, High Pressure

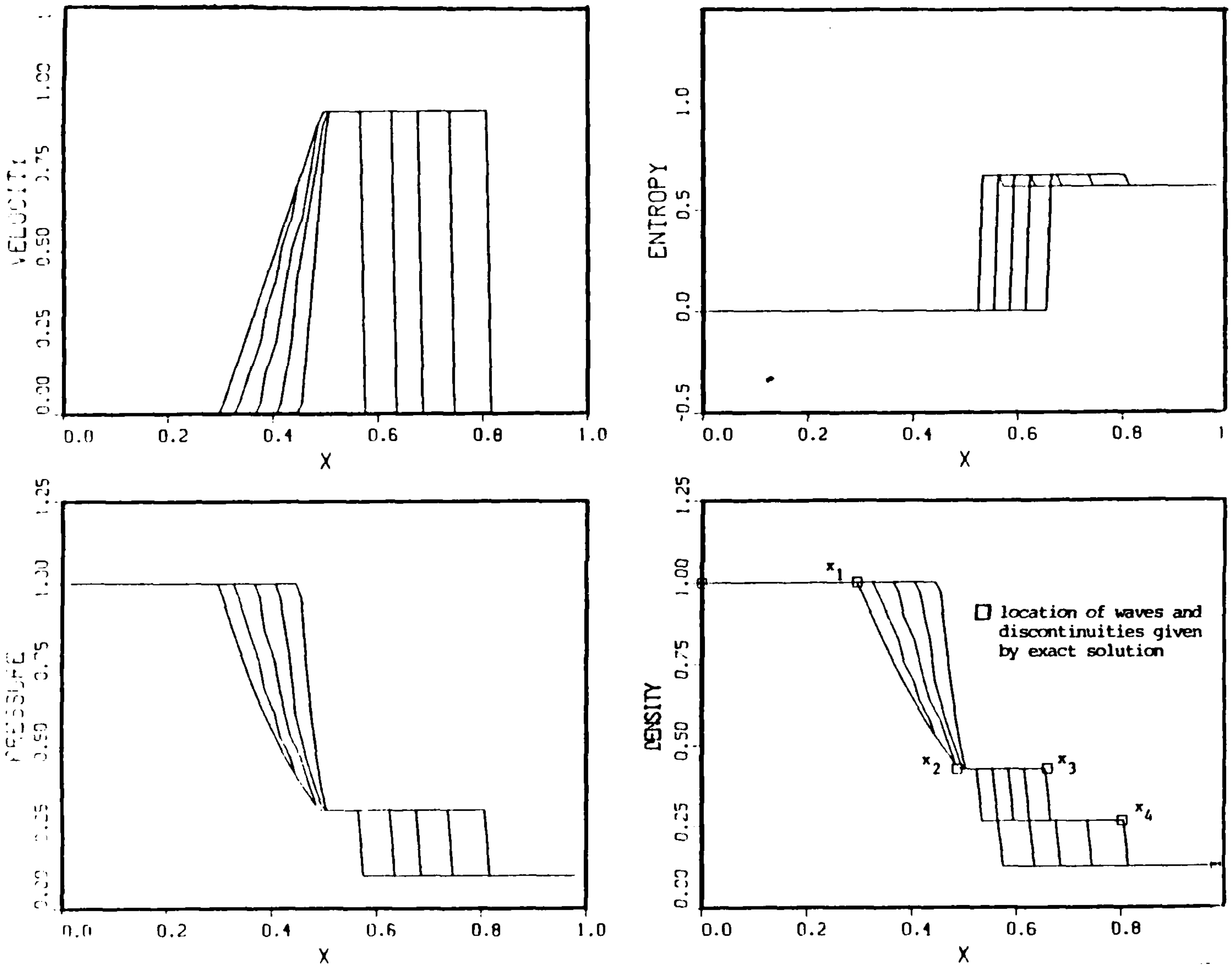
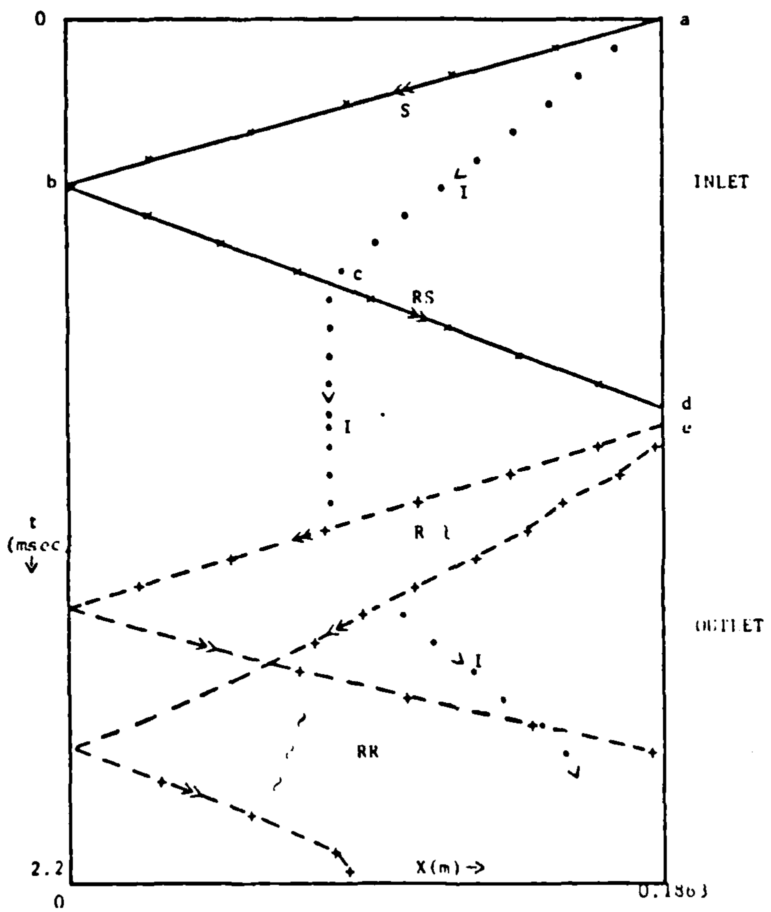


Fig. 5 : Test Case for 1-D Random Choice Method



Wave Diagram Computed by 1-D Random Choice Method. S--Shock; RS--Reflected Shock; R--Rarefaction Fan; RR--Reflected Rarefaction; I--Interface;

Figure 6.

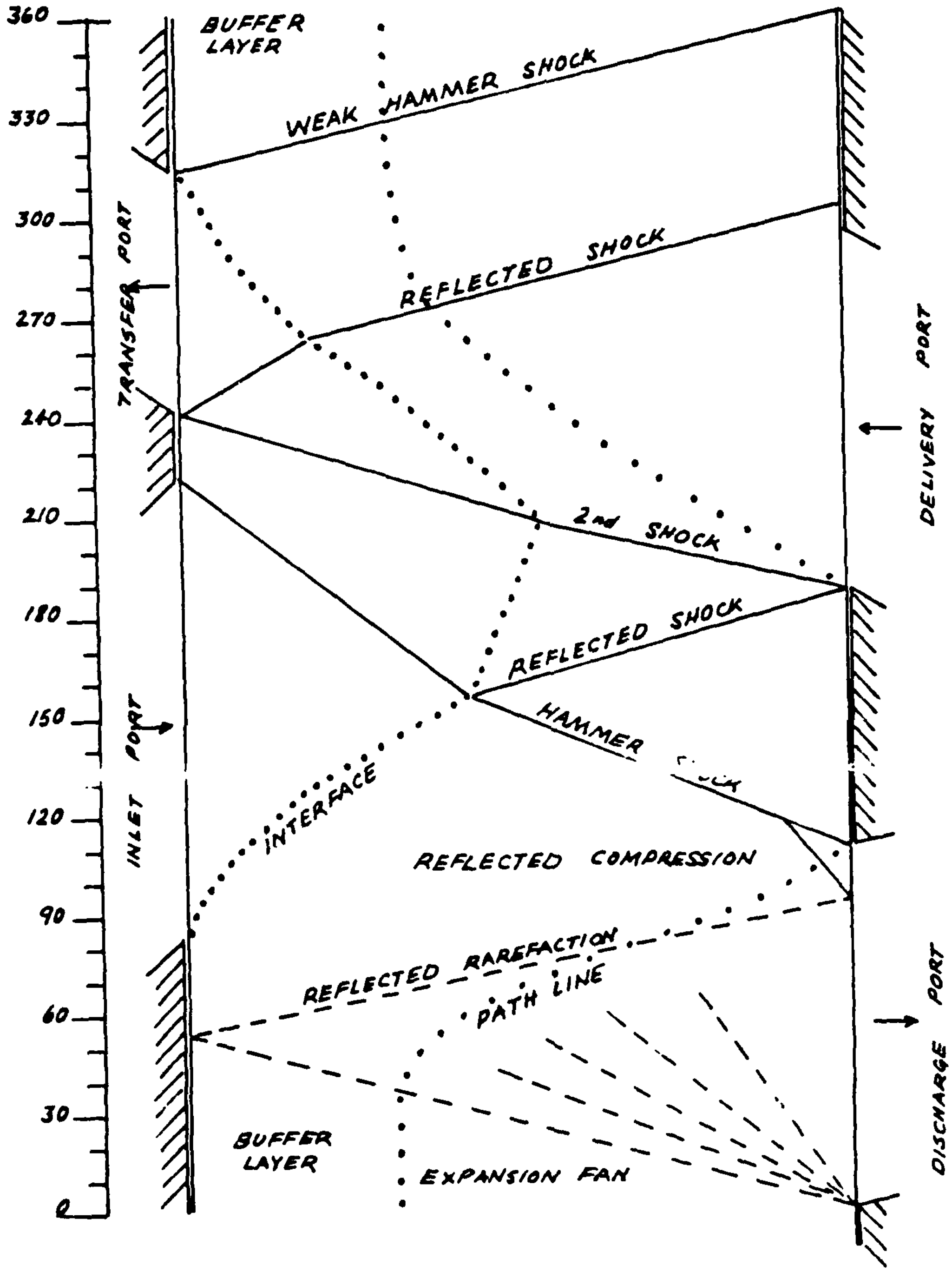


Fig.7 : Ideal Wave Diagram for General Electric Wave Engine

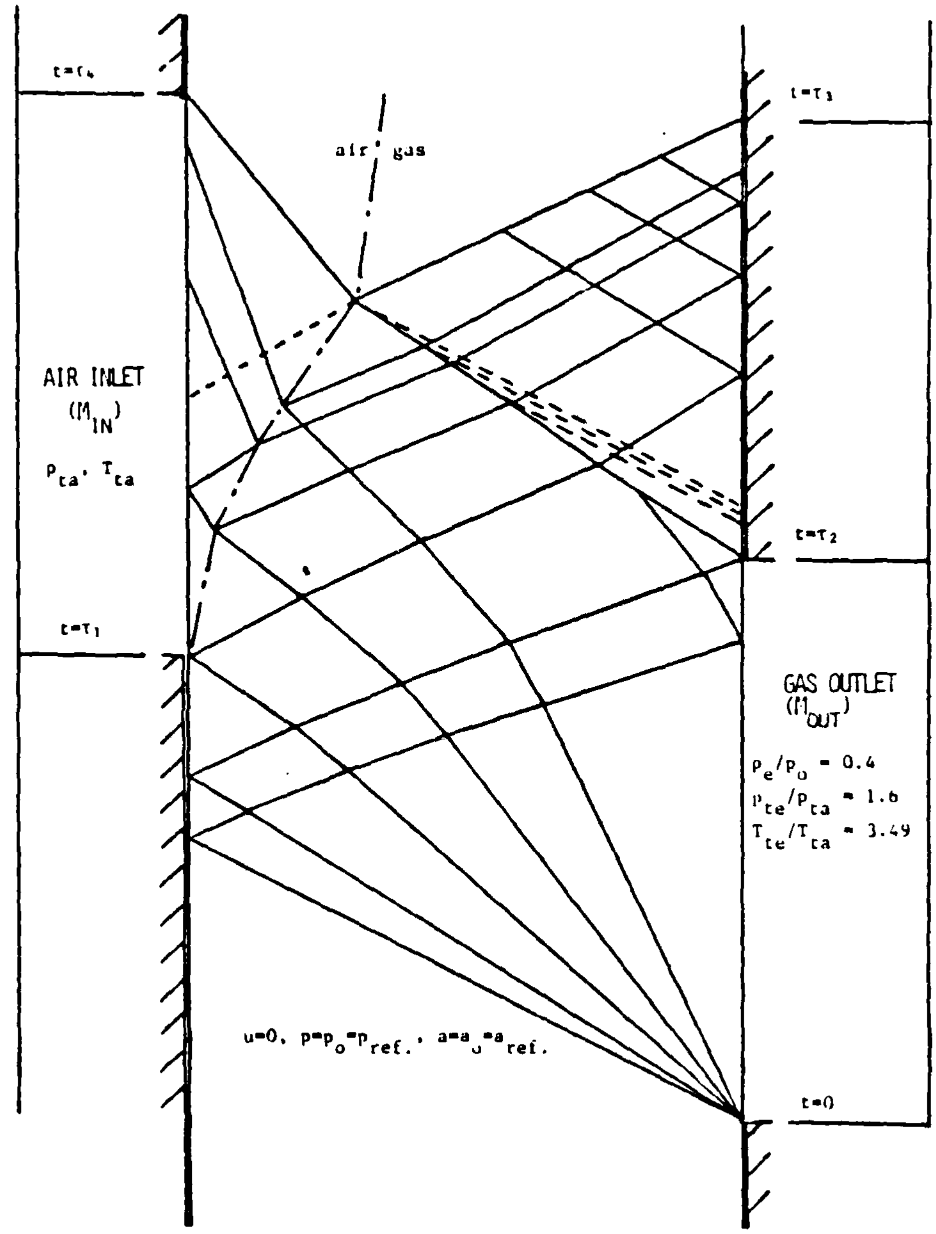


Fig.8 : Gas Exhaust and Fresh Air Induction Process in G.E. Wave Engine

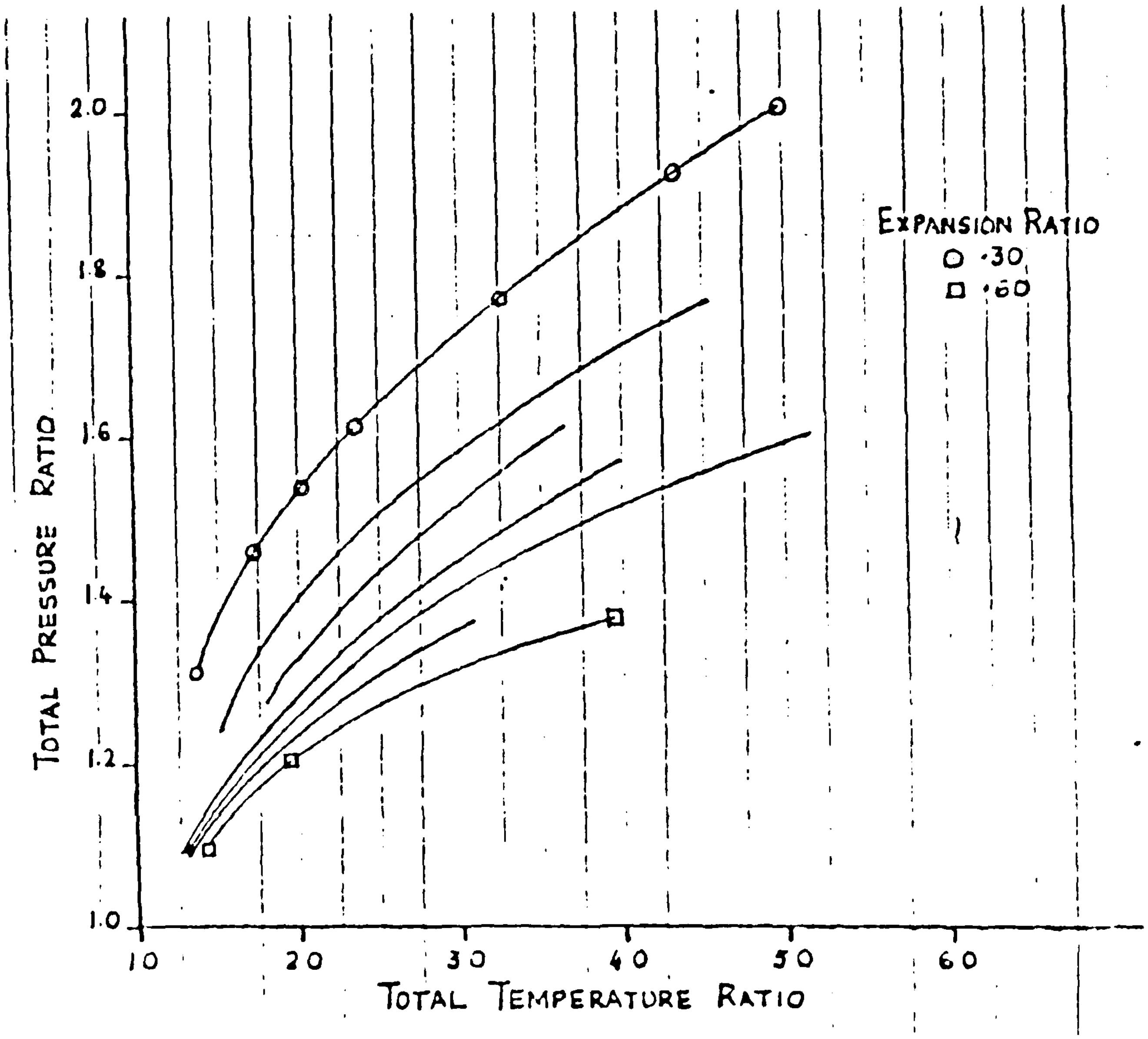


Fig.9 : Ideal Performance Curves for G.E. Wave Rotor
as Gas Generator

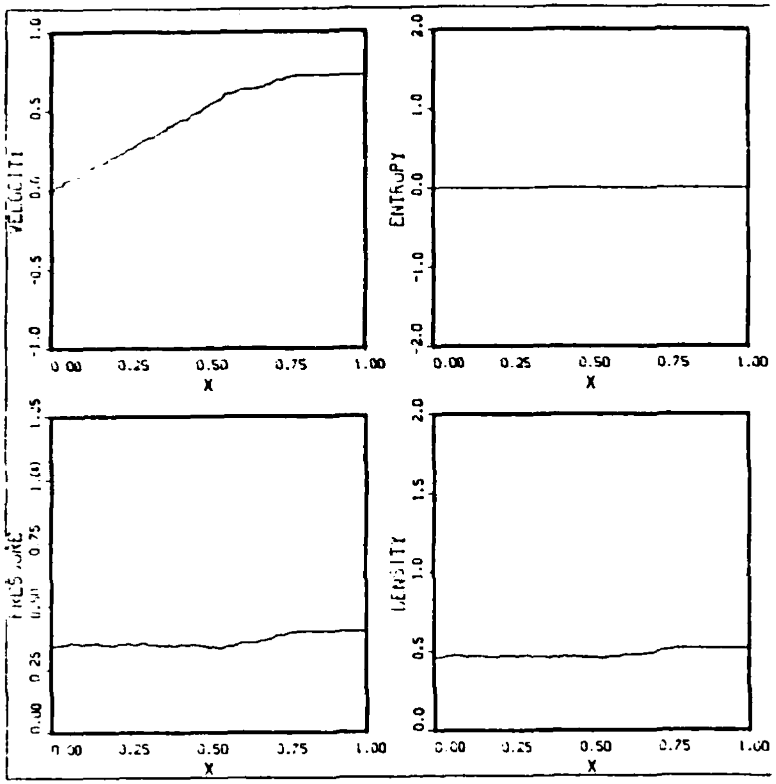


Fig.10 a : Distribution of Flow Parameters in Rotor Passage Just as Inlet Port Opens. N= 793.

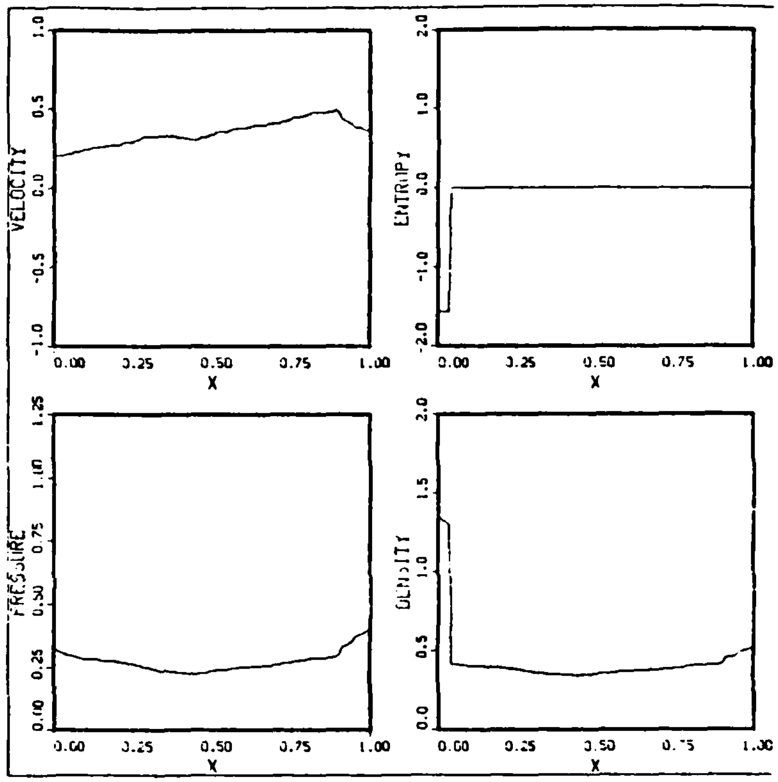


Fig.10 b : Distribution of Flow Parameters in Rotor Passage Just as Exhaust Port is Closed. N= 998.

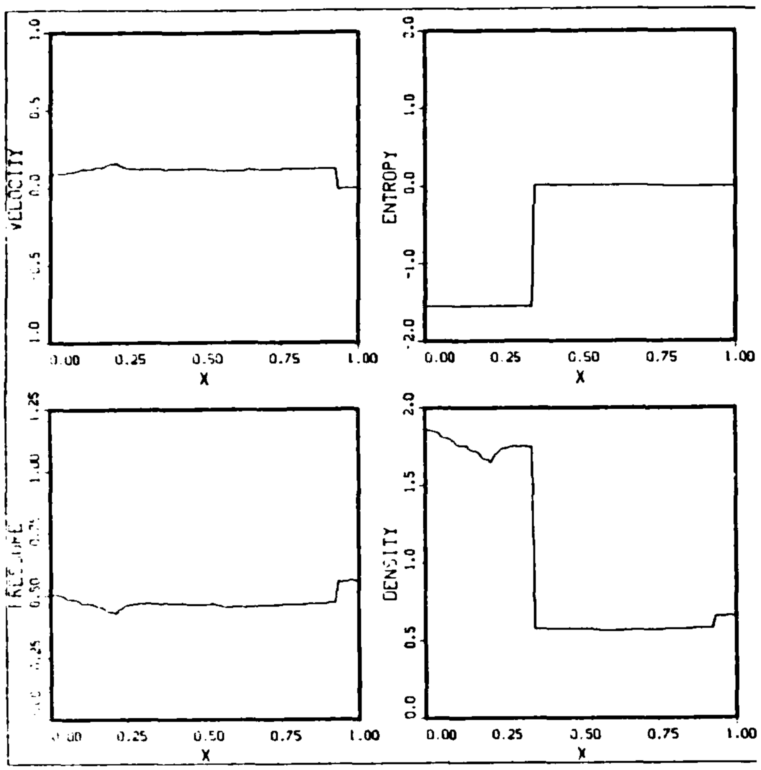


Fig.10 c : Distribution of Flow Parameters in Rotor Passage Just as Inlet Port Closes. N= 1617.

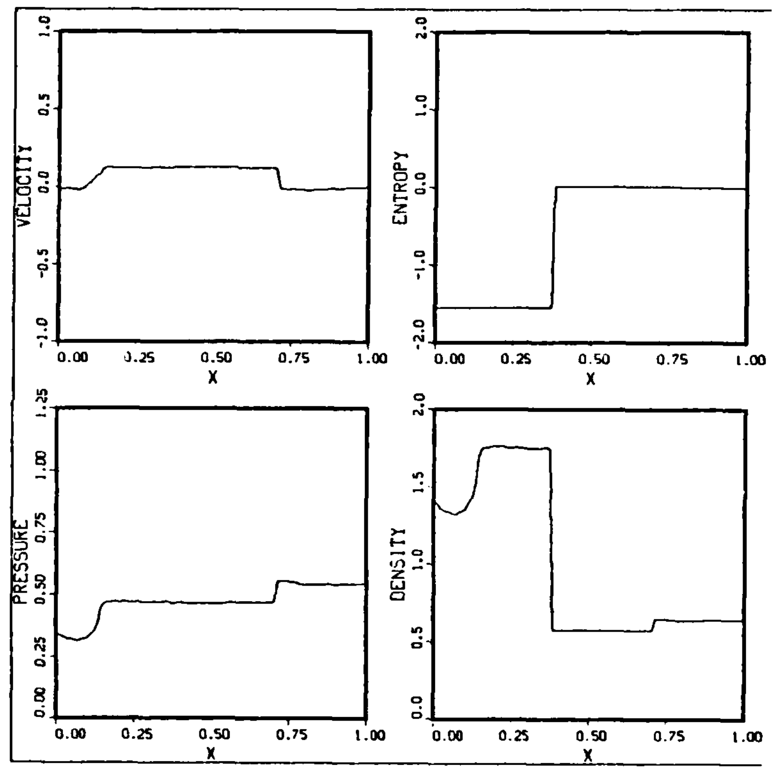


Fig.11 : Distribution of Flow Parameters in Rotor Passage When Inlet Port Closes Late, i.e. After Arrival Of Shock. N= 1700.

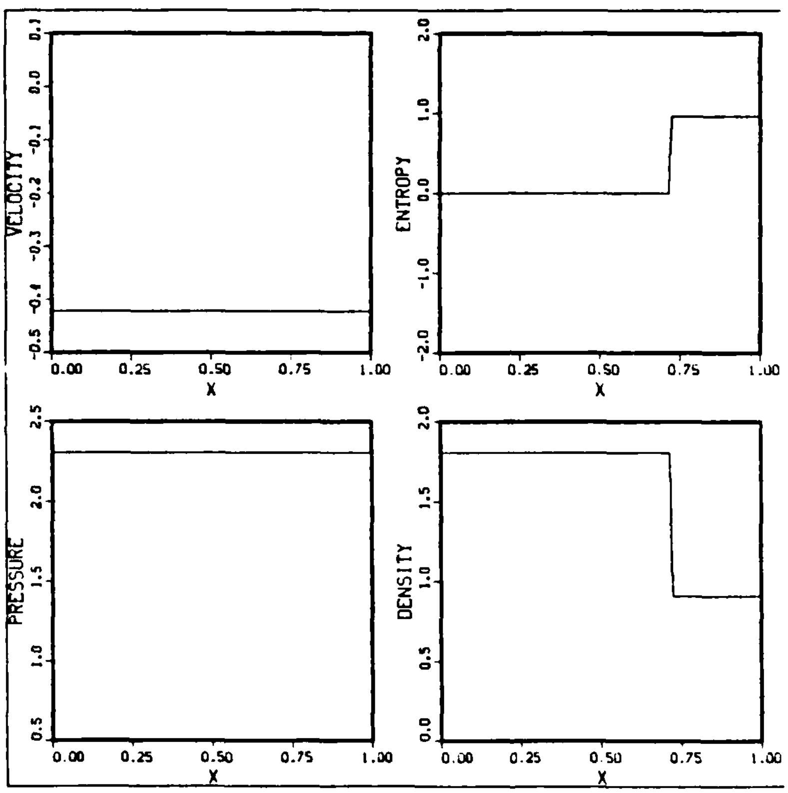
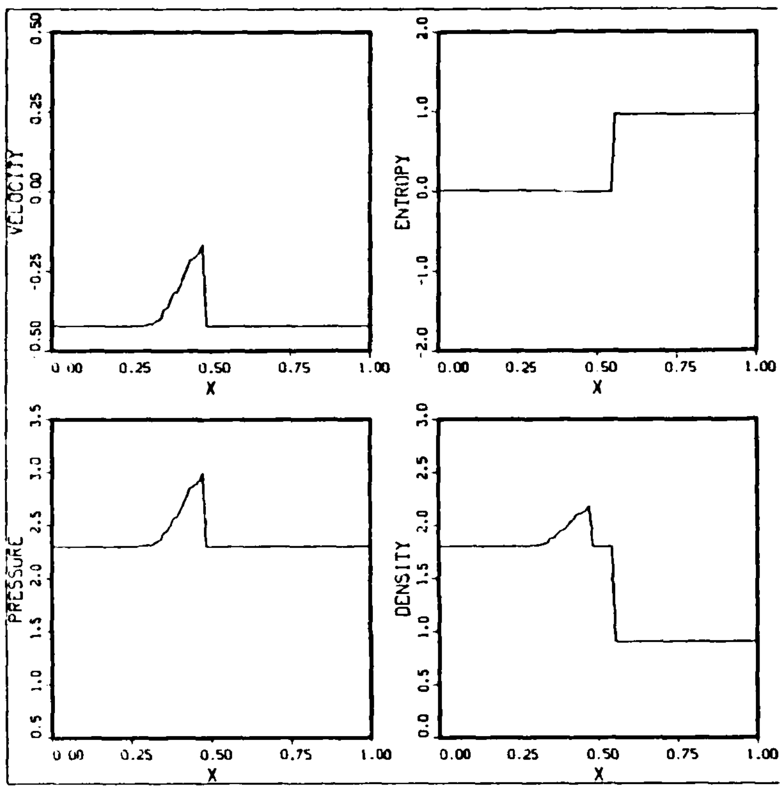
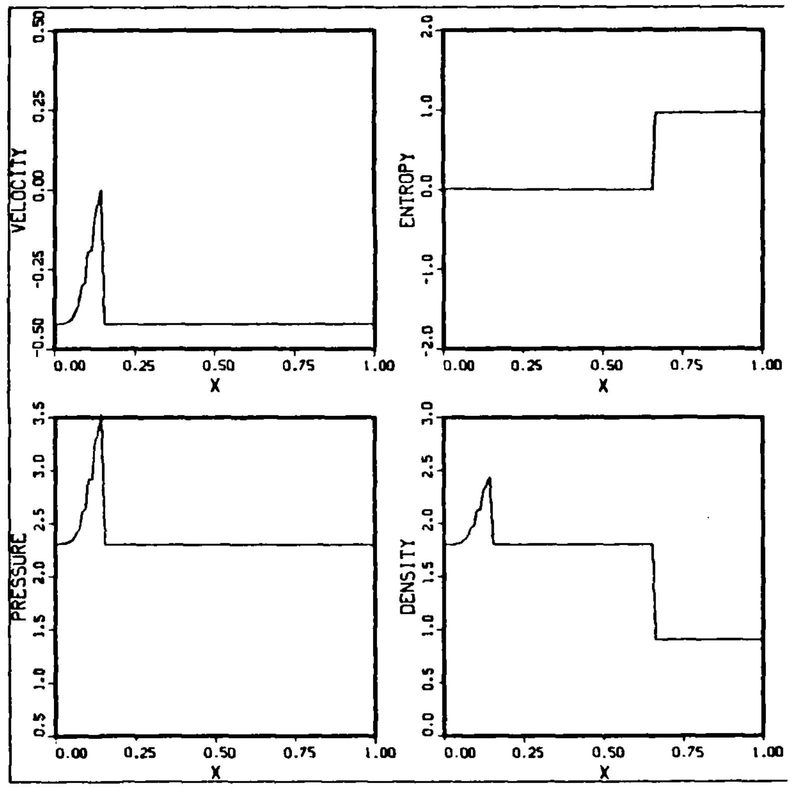


Fig.12 a : Distribution of Flow Parameters in Rotor Passage when the H.P. Air Outlet Port Opens on Time.



b : H.P. Air Outlet Port Opens Prematurely



c : H.P. Air Outlet Port Opens Late

Fig.12 b,c : Distribution of Flow Parameters in Rotor Passage for Two Conditions of Mismatch

APPENDIX A

Listing of Program RCM

```
C     PROGRAM RCM WITH VAN DER CORPUT SAMPLING AND SINGLE TIME STEP
      INTEGER QPRINT, QSTOP, SWL, SWR
      DIMENSION XX(6),YY(6)
      DIMENSION XARRAY(100)
      DIMENSION WNORM(12),IDIGT(12)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/GLIMM1/PGLIM,RGLIM,UGLIM,PL,RL,UL,PR,RR,UR,AL,AR,GL,GR,EPS
      COMMON/GLIMM2/DT,DX,XI
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      COMMON/SAMPLE/WNORM,IDIGT
      COMMON XARRAY,N1
      CALL COMPRS
      CALL BLOWUP(0.5)
      CALL PAGE(11.0,8.5)
      CALL HWSCAL('SCREEN')
      DATA K,SWL,SWR/500,1,3/
      DATA N,CFLNUM,TTOTAL/0,0.60,0.0/
      DATA PSEXIT,PSINL,PSINR,RINL,RINR/116954.,3770000.,3819952.50,6.8,
     *6.800/
      DATA PSOUT1,PSOUT2,PSOUT3/3819952.5,2431800.0,1530007.5/
      DATA PTOTIN,RTOTIN/1656663.8,7.486/
      DATA PREF,RREF,XREF/1656663.8,7.486,0.1800/
      G=1.4
      GL=1.4
      GR=1.4
      EPS=1.E-06
      QSTOP=20
      N1=0
      JCOUNT=0
      KCOUNT=0
      UEXMAX=0.
      DX=0.01
      AREF=SQRT(PREF/RREF)
      TIMREF=XREF/AREF
      RMU=SQRT((G-1.)/(G+1.))
      X(1)=-0.5*DX
      ZETA=WDP(1)
      XII=DX*(WDP(0)-0.5)
      DO 25 I=2,203
      X(I)=X(I-1)+0.5*DX
   25 CONTINUE
      DO 35 I=1,100
      XARRAY(I)=X(I*2+1)
   35 CONTINUE
C     INITIAL DATA
C     CALL INIT1
C     CALL INIT2L(PSEXIT)
C     CALL INIT2R(PSEXIT,PREF,RREF)
C     CALL INIT3L(PSINL,RINL)
      CALL INIT3R(PSINR,RINR)
C     NONDIMENSIONALIZATION
      DO 30 I=1,203,2
      P(I)=P(I)/PREF
```

```
      R(I)=R(I)/RREF                                                     RCM00570
      U(I)=U(I)/AREF                                                     RCM00580
      A(I)=A(I)/AREF                                                     RCM00590
      S(I)=ALOG(P(I)/R(I)**G)                                            RCM00600
   30 CONTINUE                                                           RCM00610
      CALL PLOT1(K)                                                      RCM00620
      DO 40 J=1,K                                                        RCM00630
      N=N+1                                                              RCM00640
      XII=DX*(WDP(0)-0.5)                                                RCM00650
      QPRINT=N/50                                                        RCM00660
      DT=100.                                                            RCM00670
      DO 50 I=1,203,2                                                    RCM00680
      DTT=CFLNUM*DX/(2.*AMAX1(ABS(A(I)+U(I)),ABS(A(I)-U(I))))            RCM00690
      DT=AMIN1(DTT,DT)                                                   RCM00700
   50 CONTINUE                                                           RCM00710
      TTOTAL=TTOTAL+DT                                                   RCM00720
      TIME=TTOTAL*TIMREF                                                 RCM00730
      XI=-XII                                                            RCM00740
      DO 60 I=1,201,2                                                    RCM00750
      PL=P(I)                                                            RCM00760
      RL=R(I)                                                            RCM00770
      UL=U(I)                                                            RCM00780
      PR=P(I+2)                                                          RCM00790
      RR=R(I+2)                                                          RCM00800
      UR=U(I+2)                                                          RCM00810
      XITEMP=XI                                                          RCM00820
      IF(I.EQ.1) XI=ABS(XI)                                              RCM00830
      IF((I.EQ.201).AND.(XI.GT.0.0)) XI=-XI                              RCM00840
      CALL GLIMM(QSTOP,PSTAR,USTAR,ASTAR)                                RCM00850
      XI=XITEMP                                                          RCM00860
      P(I+1)=PGLIM                                                       RCM00870
      R(I+1)=RGLIM                                                       RCM00880
      U(I+1)=UGLIM                                                       RCM00890
   60 CONTINUE                                                           RCM00900
      DO 70 I=1,201,2                                                    RCM00910
      IF(XI.LT.0.) GOTO 80                                               RCM00920
      P(I+2)=P(I+1)                                                      RCM00930
      R(I+2)=R(I+1)                                                      RCM00940
      U(I+2)=U(I+1)                                                      RCM00950
      A(I+2)=SQRT(G*P(I+2)/R(I+2))                                       RCM00960
      S(I+2)=ALOG(P(I+2)/R(I+2)**G)                                      RCM00970
      GOTO 70                                                            RCM00980
   80 P(I)=P(I+1)                                                        RCM00990
      R(I)=R(I+1)                                                        RCM01000
      U(I)=U(I+1)                                                        RCM01010
      A(I)=SQRT(G*P(I)/R(I))                                             RCM01020
      S(I)=ALOG(P(I)/R(I)**G)                                            RCM01030
   70 CONTINUE                                                           RCM01040
C     CALL GE(SWL,SWR,N,TTOTAL,TIME,UEXMAX,PTOTIN,PREF)                  RCM01050
C     CALL DETON(SWL,SWR,N,QPRINT,TTOTAL,TIME)                           RCM01060
      CALL SPCTRA(N,SWL,SWR,TIME,UEXMAX,PSEXIT,PSOUT1,PSOUT2,PSOUT3,J    RCM01070
     *COUNT,QPRINT,TTOTAL,KCOUNT)                                        RCM01080
      IF(SWL.EQ.1) CALL BCL1                                             RCM01090
      IF(SWL.EQ.2) CALL BCL2(PSEXIT,PREF)                                RCM01100
```

```
      IF(SWL.EQ.3) CALL BCL3(PSINL,RINL,PREF,RREF)
      IF(SWL.EQ.4) CALL BCL4(PTOTIN,RTOTIN,PREF,RREF)
      IF(SWL.EQ.5) CALL BCL5
      IF(SWR.EQ.1) CALL BCR1
      IF(SWR.EQ.2) CALL BCR2(PSEXIT,PREF)
      IF(SWR.EQ.3) CALL BCR3(PSINR,RINR,PREF,RREF)
      IF(SWR.EQ.4) CALL BCR4(PTOTIN,RTOTIN,PREF,RREF)
      IF(SWR.EQ.5) CALL BCR5
      IF ((N.EQ.(50*QPRINT)).AND.(N.GE.0)) CALL PLOT2(N,K)
   40 CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
      STOP
      END
      SUBROUTINE GLIMM(QSTOP,PSTAR,USTAR,ASTAR)
      INTEGER Q,QSTOP
      REAL ML,MR,MLN,MRN
      COMMON/GLIMM1/PGLIM,RGLIM,UGLIM,PL,RL,UL,PR,RR,UR,AL,AR,GL,GR,EPS
      COMMON/GLIMM2/DT,DX,XI
      DATA Q,ML,MR/0,100.,100./
      PSTAR=0.5*(PL+PR)
      COEFL=SQRT(PL*RL)
      COEFR=SQRT(PR*RR)
      ALPHA=1.
C     BEGIN GODUNOV ITERATION
   30 Q=Q+1
      IF(PSTAR.LT.EPS) PSTAR=EPS
C     COMPUTE NEXT ITERATION FOR ML AND MR
      MLN=COEFL*PHI(PSTAR,PL)
      MRN=COEFR*PHI(PSTAR,PR)
      DIFML=ABS(MLN-ML)
      DIFMR=ABS(MRN-MR)
      ML=MLN
      MR=MRN
C     COMPUTE NEW PSTAR
      PTIL=PSTAR
      PSTAR=(UL-UR+PL/ML+PR/MR)/(1./ML+1./MR)
      PSTAR=ALPHA*PSTAR+(1.-ALPHA)*PTIL
      IF(Q.LE.QSTOP) GOTO 10
      IF(ABS(PSTAR-PTIL).LT.EPS) GOTO 20
C     COMPUTE NEW ALPHA
      ALPHA=0.5*ALPHA
      Q=0
      IF((1.-ALPHA).LT.EPS) GOTO 20
   10 IF(DIFML.GE.EPS) GOTO 30
      IF(DIFMR.GE.EPS) GOTO 30
C     END OF GODUNOV ITERATION; COMPUTE USTAR
   20 USTAR=(PL-PR+ML*UL+MR*UR)/(ML+MR)
C     BEGIN SAMPLING PROCEDURE
      IF (XI.LT.USTAR*DT) GO TO 40
C     RIGHT SIDE; SELECT CASE OF SHOCK OR EXPANSION
      IF (PSTAR.LT.PR) GO TO 50
C     RIGHT WAVE IS A SHOCK WAVE
      WR=UR+MR/RR
```

```
      IF (XI.LT.WR*DT) GO TO 60
C     RIGHT OF RIGHT SHOCK CASE
      RGLIM=RR
      PGLIM=PR
      UGLIM=UR
      RETURN
C     LEFT OF RIGHT SHOCK CASE
   60 RGLIM=-MR/(USTAR-WR)
      PGLIM=PSTAR
      UGLIM=USTAR
      RETURN
C     RIGHT WAVE IS A RAREFACTION WAVE
   50 CONST=PR/RR**GR
      RSTAR=(PSTAR/CONST)**(1./GR)
      ASTAR=SQRT(GR*PSTAR/RSTAR)
      AR=SQRT(GR*PR/RR)
      IF (XI.GE.(USTAR+ASTAR)*DT) GO TO 70
C     LEFT OF RIGHT FAN CASE
      RGLIM=RSTAR
      UGLIM=USTAR
      PGLIM=PSTAR
      RETURN
C     SELECT RIGHT OF FAN OR IN FAN
   70 IF (XI.GE.(UR+AR)*DT) GO TO 80
C     IN RIGHT FAN CASE
      UGLIM=2./(GR+1.)*(XI/DT-AR+(GR-1.)/2.*UR)
      RGLIM=(((AR+(GR-1.)/2.*(UGLIM-UR))**2.)/(GR*CONST))**(1./(GR-1.))
      PGLIM=CONST*RGLIM**GR
      RETURN
C     RIGHT OF RIGHT FAN CASE
   80 RGLIM=RR
      PGLIM=PR
      UGLIM=UR
      RETURN
C     LEFT SIDE; SELECT CASE OF SHOCK OR RAREFACTION
   40 IF (PSTAR.LT.PL) GO TO 90
C     LEFT WAVE IS A SHOCK WAVE
      WL=UL-ML/RL
      IF (XI.GE.WL*DT) GO TO 100
C     LEFT OF LEFT SHOCK CASE
      RGLIM=RL
      PGLIM=PL
      UGLIM=UL
      RETURN
C     RIGHT OF LEFT SHOCK CASE
  100 RGLIM=ML/(USTAR-WL)
      PGLIM=PSTAR
      UGLIM=USTAR
      RETURN
C     LEFT WAVE IS A RAREFACTION WAVE
   90 CONST=PL/RL**GL
      RSTAR=(PSTAR/CONST)**(1./GL)
      ASTAR=SQRT(GL*PSTAR/RSTAR)
      AL=SQRT(GL*PL/RL)
```

```
      IF (XI.LT.(USTAR-ASTAR)*DT) GO TO 110                             RCM02190
C     RIGHT OF LEFT FAN CASE                                            RCM02200
      RGLIM=RSTAR                                                       RCM02210
      PGLIM=PSTAR                                                       RCM02220
      UGLIM=USTAR                                                       RCM02230
      RETURN                                                            RCM02240
C     SELECT LEFT OF FAN OR IN FAN CASE                                 RCM02250
  110 IF (XI.LT.(UL-AL)*DT) GO TO 120                                   RCM02260
C     IN LEFT FAN CASE                                                  RCM02270
      UGLIM=2./(GL+1.)*(AL+(GL-1.)/2.*UL+XI/DT)                         RCM02280
      RGLIM=(((AL+(GL-1.)/2.*(UL-UGLIM))**2.)/(GL*CONST))**(1./(GL-1.)) RCM02290
      PGLIM=CONST*RGLIM**GL                                             RCM02300
      RETURN                                                            RCM02310
C     LEFT OF LEFT FAN CASE                                             RCM02320
  120 RGLIM=RL                                                          RCM02330
      PGLIM=PL                                                          RCM02340
      UGLIM=UL                                                          RCM02350
      RETURN                                                            RCM02360
      END                                                               RCM02370
      FUNCTION PHI(Y,Z)                                                 RCM02380
      REAL RMU                                                          RCM02390
      COMMON/FUN1/G,PA,RA,UA,RB,RMU                                     RCM02400
      EPS=1.E-06                                                        RCM02410
      PARAM=Y/Z                                                         RCM02420
      IF (ABS(1.-PARAM).GE.EPS) GO TO 10                                RCM02430
      PHI=SQRT(G)                                                       RCM02440
      RETURN                                                            RCM02450
   10 IF (PARAM.GE.1.) GO TO 20                                         RCM02460
      PHI=(G-1.)/2.*(1.-PARAM)/(SQRT(G)*(1.-PARAM**((G-1.)/(2.*G))))    RCM02470
      RETURN                                                            RCM02480
   20 PHI=SQRT((G+1.)/2.*PARAM+(G-1.)/2.)                               RCM02490
      RETURN                                                            RCM02500
      END                                                               RCM02510
      FUNCTION PHI1(PB)                                                 RCM02520
      REAL RMU                                                          RCM02530
      COMMON/FUN1/G,PA,RA,UA,RB,RMU                                     RCM02540
      PHI1=(PB-PA)*SQRT((1.-RMU**2.)/(RA*(PB+RMU**2.*PA)))              RCM02550
      RETURN                                                            RCM02560
      END                                                               RCM02570
      FUNCTION PSI(PB)                                                  RCM02580
      REAL RMU                                                          RCM02590
      COMMON/FUN1/G,PA,RA,UA,RB,RMU                                     RCM02600
      PSI=SQRT(1.-RMU**4.)/RMU**2./SQRT(RA)*PA**(1./(2.*G))*(PB**((G-1.)RCM02610
     */(2.*G))-PA**((G-1.)/(2.*G)))                                     RCM02620
      RETURN                                                            RCM02630
      END                                                               RCM02640
      SUBROUTINE INIT1                                                  RCM02650
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)               RCM02660
      COMMON/FUN1/G,PA,RA,UA,RB,RMU                                     RCM02670
      COMMON/SUBS/P,R,U,A,S,X                                           RCM02680
      DO 10 I=1,9,2                                                     RCM02690
      P(I)=810600.00                                                    RCM02700
      R(I)=0.7132                                                       RCM02710
      U(I)=644.4                                                        RCM02720
```

```
      A(I)=SQRT(G*P(I)/R(I))
   10 CONTINUE
      DO 20 I=11,203,2
      P(I)=101325.0
      R(I)=1.22
      U(I)=0.0
      A(I)=SQRT(G*P(I)/R(I))
   20 CONTINUE
      RETURN
      END
      SUBROUTINE INIT2R(PSEXIT,PREF,RREF)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      COMMON/SUBS/P,R,U,A,S,X
      DO 10 I=3,201,2
      P(I)=PREF
      R(I)=RREF
      U(I)=0.0
      A(I)=SQRT(P(I)*G/R(I))
   10 CONTINUE
      P(1)=P(3)
      R(1)=R(3)
      U(1)=-U(3)
      A(1)=SQRT(G*P(1)/R(1))
      P(203)=PSEXIT
      R(203)=R(201)
      PA=P(201)
      RA=R(201)
      UA=U(201)
      PB=P(203)
      RB=R(203)
      IF(PA.GT.PB) GO TO 20
      U(203)=UA-PHI1(PB)
      GO TO 30
   20 U(203)=UA-PSI(PB)
   30 A(203)=SQRT(G*P(203)/R(203))
      RETURN
      END
      SUBROUTINE INIT2L(PSEXIT)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      COMMON/SUBS/P,R,U,A,S,X
      DO 10 I=3,201,2
      P(I)=285080.0
      R(I)=0.897
      U(I)=0.0
      A(I)=SQRT(G*P(I)/R(I))
   10 CONTINUE
      RETURN
      END
      SUBROUTINE INIT3L(PSINL,RINL)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      COMMON/SUBS/P,R,U,A,S,X
```

```
      DO 10 I=3,201,2
      P(I)=2390000.0
      R(I)=9.787
      U(I)=0.0
      A(I)=SQRT(G*P(I)/R(I))
   10 CONTINUE
      P(1)=PSINL
      R(1)=RINL
      PA=P(3)
      RA=R(3)
      UA=U(3)
      PB=P(1)
      U(1)=UA+PHI1(PB)
      A(1)=SQRT(G*P(1)/R(1))
      P(203)=P(201)
      R(203)=R(201)
      U(203)=-U(201)
      A(203)=SQRT(G*P(203)/R(203))
      RETURN
      END
      SUBROUTINE INIT3R(PSINR,RINR)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      DO 10 I=3,201,2
      P(I)=2421667.5
      R(I)=9.787
      U(I)=0.0
      A(I)=SQRT(G*P(I)/R(I))
   10 CONTINUE
      P(203)=PSINR
      PA=P(201)
      RA=R(201)
      UA=U(201)
      PB=P(203)
      U(203)=UA-PHI1(PB)
      R(203)=RINR
      A(203)=SQRT(G*P(203)/R(203))
      P(1)=P(3)
      R(1)=R(3)
      U(1)=-U(3)
      A(1)=SQRT(G*P(1)/R(1))
      RETURN
      END
      SUBROUTINE BCL1
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      P(1)=P(3)
      R(1)=R(3)
      U(1)=-U(3)
      A(1)=SQRT(G*P(1)/R(1))
      RETURN
      END
```

```
      SUBROUTINE BCR1
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      P(203)=P(201)
      R(203)=R(201)
      U(203)=-U(201)
      A(203)=SQRT(G*P(203)/R(203))
      RETURN
      END
      SUBROUTINE BCL2(PSEXIT,PREF)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      P(1)=PSEXIT/PREF
      R(1)=R(3)
      U(1)=U(3)
      A(1)=SQRT(G*P(1)/R(1))
      RETURN
      END
      SUBROUTINE BCR2(PSEXIT,PREF)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      P(203)=PSEXIT/PREF
      R(203)=R(201)
      PA=P(201)
      RA=R(201)
      UA=U(201)
      PB=P(203)
      RB=R(203)
      IF(PA.GT.PB) GO TO 10
      U(203)=UA-PHI1(PB)
      GO TO 20
   10 U(203)=UA-PSI(PB)
   20 A(203)=SQRT(G*P(203)/R(203))
      RETURN
      END
      SUBROUTINE BCL3(PSINL,RINL,PREF,RREF)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      P(1)=PSINL/PREF
      R(1)=RINL/RREF
      PA=P(3)
      RA=R(3)
      UA=U(3)
      PB=P(1)
      U(1)=UA+PHI1(PB)
      A(1)=SQRT(G*P(1)/R(1))
      RETURN
      END
      SUBROUTINE BCR3(PSINR,RINR,PREF,RREF)
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
```

```
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      P(203)=PSINR/PREF
      R(203)=RINR/RREF
      PA=P(201)
      RA=R(201)
      UA=U(201)
      PB=P(203)
      U(203)=UA-PHI1(PB)
      A(203)=SQRT(G*P(203)/R(203))
      RETURN
      END
      SUBROUTINE BCL4 (PTOTIN,RTOTIN,PREF,RREF)
      INTEGER QOUT
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      DIMENSION XARRAY(100)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      COMMON XARRAY,N1
      N1=N1+1
      QOUT=N1/5
      PTOT=PTOTIN/PREF
      RTOT=RTOTIN/RREF
      ATOT=SQRT(G*PTOT/RTOT)
      STOT=ALOG(PTOT/RTOT**G)
      U(1)=U(3)
      A(1)=SQRT(ATOT**2.-(G-1.)/2.*ABS(U(1))**2.)
      AMACH=U(1)/A(1)
      IF(AMACH.LT.0.0) GO TO 60
      P(1)=PTOT/(1.+(G-1.)/2.*AMACH**2.)**(G/(G-1.))
      R(1)=RTOT/(1.+(G-1.)/2.*AMACH**2.)**(1./(G-1.))
      S(1)=ALOG(P(1)/R(1)**G)
      GO TO 50
   60 P(1)=PTOT/(1.+(G-1.)/2.*ABS(AMACH)**2.)**(G/(G-1.))
      R(1)=RTOT/(1.+(G-1.)/2.*ABS(AMACH)**2.)**(1./(G-1.))
      S(1)=ALOG(P(1)/R(1)**G)
   50 RETURN
      END
      SUBROUTINE BCR4 (PTOTIN,RTOTIN,PREF,RREF)
      INTEGER QOUT
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      DIMENSION XARRAY(100)
      COMMON/SUBS/P,R,U,A,S,X
      COMMON/FUN1/G,PA,RA,UA,RB,RMU
      COMMON XARRAY,N1
      N1=N1+1
      QOUT=N1/25
      PTOT=PTOTIN/PREF
      RTOT=RTOTIN/RREF
      ATOT=SQRT(G*PTOT/RTOT)
      STOT=ALOG(PTOT/RTOT**G)
      U(203)=U(201)
      A(203)=SQRT(ATOT**2.-(G-1.)/2.*ABS(U(203))**2.)
      AMACH=U(203)/A(203)
```

```
      IF(AMACH.LT.0.0) GO TO 60
      P(203)=PTOT/(1.+(G-1.)/2.*AMACH**2.)**(G/(G-1.))
      R(203)=RTOT/(1.+(G-1.)/2.*AMACH**2.)**(1./(G-1.))
      S(203)=ALOG(P(203)/R(203)**G)
      GO TO 50
   60 P(203)=PTOT/(1.+(G-1.)/2.*ABS(AMACH)**2.)**(G/(G-1.))
      R(203)=RTOT/(1.+(G-1.)/2.*ABS(AMACH)**2.)**(1./(G-1.))
      S(203)=ALOG(P(203)/R(203)**G)
   50 RETURN
      END
      SUBROUTINE BCL5
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      P(1)=P(3)
      R(1)=R(3)
      U(1)=U(3)
      A(1)=A(3)
      RETURN
      END
      SUBROUTINE BCR5
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
      P(203)=P(201)
      R(203)=R(201)
      U(203)=U(201)
      A(203)=A(201)
      RETURN
      END
      FUNCTION WDP(II)
      DIMENSION WNORM(12),IDIGT(12)
      COMMON/SAMPLE/WNORM,IDIGT
      IF (II.EQ.0) GO TO 10
      L1=2
      L2=1
      DO 20 JJ=1,12
      IDIGT(JJ)=0
      WNORM(JJ)=1./FLOAT(L1**JJ)
   20 CONTINUE
      WDP=0.
      RETURN
   10 DO 40 JJ=1,12
      L1=2
      L2=1
      KJ0=IDIGT(JJ)
      KJN=MOD((KJ0+1),L1)
      IDIGT(JJ)=KJN
      IF (KJ0.LT.KJN) GO TO 50
   40 CONTINUE
   50 SUM=0.
      DO 60 JJ=1,12
      KNEW=MOD(IDIGT(JJ)*L2,L1)
      SUM=SUM+FLOAT(KNEW)*WNORM(JJ)
   60 CONTINUE
      WDP=SUM
```

```
      RETURN                                                            RCM05430
      END                                                               RCM05440
      SUBROUTINE PLOT1(K)                                               RCM05450
      DIMENSION XORG(4),YORG(4),YMAX(4),YMIN(4)                         RCM05460
      DATA XORG/0.5,4.75,0.5,4.75/                                      RCM05470
      DATA YORG/0.5,0.5,4.75,4.75/                                      RCM05480
      DATA YMAX/3.50,3.0,0.5,2.0/                                       RCM05490
      DATA YMIN/0.5,0.0,-0.5,-2.0/                                      RCM05500
      DO 10 I=1,4                                                       RCM05510
      CALL PHYSOR(XORG(I),YORG(I))                                      RCM05520
      CALL AREA2D(3.5,3.5)                                              RCM05530
      CALL FRAME                                                        RCM05540
      CALL GRAF(0.,'SCALE',1.0,YMIN(I),'SCALE',YMAX(I))                 RCM05550
      CALL ENDGR(0)                                                     RCM05560
   10 CONTINUE                                                          RCM05570
      CALL PHYSOR(8.5,0.5)                                              RCM05580
      CALL AREA2D(2.25,7.75)                                            RCM05590
      CALL FRAME                                                        RCM05600
      CALL GRAF(0.,'SCALE',1.,0,'SCALE',K)                              RCM05610
      CALL ENDGR(0)                                                     RCM05620
      RETURN                                                            RCM05630
      END                                                               RCM05640
      SUBROUTINE PLOT2(N,K)                                             RCM05650
      DIMENSION XORG(4),YORG(4),YMAX(4),YMIN(4),KNT(4),IYNAM(10)        RCM05660
      DIMENSION PARRAY(100),RARRAY(100),UARRAY(100),SARRAY(100),XARRAY(1RCM05670
     *00)                                                               RCM05680
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)               RCM05690
      COMMON/SUBS/P,R,U,A,S,X                                           RCM05700
      COMMON XARRAY                                                     RCM05710
      DATA XORG/0.5,4.75,0.5,4.75/                                      RCM05720
      DATA YORG/0.5,0.5,4.75,4.75/                                      RCM05730
      DATA YMAX/3.50,3.0,0.5,2.0/                                       RCM05740
      DATA YMIN/0.5,0.0,-0.5,-2.0/                                      RCM05750
      DATA KNT/1,4,6,9/                                                 RCM05760
      DATA IYNAM/'PRES','SURE','$   ','DENS','ITY$','VELO','CITY','$   'RCM05770
     *,'ENTR','OPY$'/                                                   RCM05780
      DO 200 I=1,100                                                    RCM05790
      PARRAY(I)=P(I*2+1)                                                RCM05800
      RARRAY(I)=R(I*2+1)                                                RCM05810
      UARRAY(I)=U(I*2+1)                                                RCM05820
      SARRAY(I)=S(I*2+1)                                                RCM05830
  200 CONTINUE                                                          RCM05840
      DO 300 I=1,4                                                      RCM05850
      CALL PHYSOR(XORG(I),YORG(I))                                      RCM05860
      CALL AREA2D(3.5,3.5)                                              RCM05870
      CALL XNAME('X',1)                                                 RCM05880
      CALL YNAME(IYNAM(KNT(I)),100)                                     RCM05890
      CALL GRAF(0.,'SCALE',1.0,YMIN(I),'SCALE',YMAX(I))                 RCM05900
      IF(I.EQ.1) CALL SETCLR('YELLOW')                                  RCM05910
      IF(I.EQ.2) CALL SETCLR('CYAN')                                    RCM05920
      IF(I.EQ.3) CALL SETCLR('RED')                                     RCM05930
      IF(I.EQ.4) CALL SETCLR('MAGENTA')                                 RCM05940
      IF(N.EQ.K) CALL SETCLR('WHITE')                                   RCM05950
      IF(I.EQ.1) CALL CURVE (XARRAY,PARRAY,100,0)                       RCM05960
```

```
      IF(I.EQ.2) CALL CURVE (XARRAY,RARRAY,100,0)
      IF(I.EQ.3) CALL CURVE (XARRAY,UARRAY,100,0)
      IF(I.EQ.4) CALL CURVE (XARRAY,SARRAY,100,0)
      CALL ENDGR(0)
  300 CONTINUE
      RETURN
      END
      SUBROUTINE GE(SWL,SWR,N,TTOTAL,TIME,UEXMAX,PTOTIN,PREF)
      INTEGER SWL,SWR
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
C**CALCULATION STARTS AT EXHAUST PORT OPENING. SUBROUTINE STRUCTURED
C**ACCORDINGLY.
      IF((SWL.EQ.1).AND.(SWR.EQ.2)) GO TO 10
      IF((SWL.EQ.4).AND.(SWR.EQ.2)) GO TO 30
      IF((SWL.EQ.4).AND.(SWR.EQ.1)) GO TO 50
      IF((SWL.EQ.1).AND.(SWR.EQ.1)) RETURN
   10 PWALL=P(2)
      IF(PWALL.LE.(PTOTIN/PREF)) GO TO 20
      RETURN
   20 SWL=4
      WRITE(6,74)
      WRITE(6,75) N,TTOTAL,TIME
      RETURN
   30 UEXIT=U(202)
      IF(UEXMAX.LT.UEXIT) UEXMAX=UEXIT
      IF(UEXIT.LT.UEXMAX/2.) GO TO 40
      RETURN
   40 SWR=1
      WRITE(6,76)
      WRITE(6,75) N,TTOTAL,TIME
      RETURN
   50 P1SHOK=P(2)
      IF(P1SHOK.GT.PTOTIN/PREF) GO TO 60
      RETURN
   60 SWL=1
      WRITE(6,77)
      WRITE(6,75) N,TTOTAL,TIME
   74 FORMAT(5X,'INLET PORT OPENS AT:')
   75 FORMAT(5X,I4,5X,2F14.7)
   76 FORMAT(5X,'EXHAUST PORT CLOSES AT:')
   77 FORMAT(5X,'INLET PORT CLOSES AT:')
      RETURN
      END
      SUBROUTINE SPCTRA(N,SWL,SWR,TIME,UEXMAX,PSEXIT,PSOUT1,PSOUT2,PSOUT
     *3,JCOUNT,QPRINT,TTOTAL,KCOUNT)
      INTEGER SWL,SWR,QPRINT
      DIMENSION P(203),R(203),U(203),A(203),S(203),X(203)
      COMMON/SUBS/P,R,U,A,S,X
C*CALCULATION STARTS AT HP GAS IN PORT. JCOUNT IS NUMBERED ACCORDINGLY**
      IF((SWL.EQ.1).AND.(SWR.EQ.3)) GO TO 10
      IF((SWL.EQ.2).AND.(SWR.EQ.3)) GO TO 20
      IF((SWL.EQ.2).AND.(SWR.EQ.1)) GO TO 30
      IF((SWL.EQ.5).AND.(SWR.EQ.1)) GO TO 40
```

```
      IF((SWL.EQ.2).AND.(SWR.EQ.5)) GO TO 50                            RCM06510
      IF((SWL.EQ.2).AND.(SWR.EQ.4)) GO TO 60                            RCM06520
      IF((SWL.EQ.1).AND.(SWR.EQ.4)) GO TO 70                            RCM06530
   10 IF(U(3).LT.0.0) GO TO 11                                          RCM06540
      RETURN                                                            RCM06550
   11 JCOUNT=JCOUNT+1                                                   RCM06560
      PSEXIT=PSOUT1                                                     RCM06570
      SWL=2                                                             RCM06580
      WRITE(6,12)                                                       RCM06590
      WRITE(6,13)N,TIME,SWL,SWR,JCOUNT                                  RCM06600
   12 FORMAT(5X,'H.P. AIR OUT PORT OPENS AT')                           RCM06610
   13 FORMAT(5X,I4,5X,F9.7,5X,3I3)                                      RCM06620
      RETURN                                                            RCM06630
   20 DO 26 I=5,199,2                                                   RCM06640
      IF((R(I)-R(I+2)).GT.0.1) GO TO 22                                 RCM06650
      GO TO 26                                                          RCM06660
   22 XCNTCT=X(I)                                                       RCM06670
      UCNTCT=U(I)                                                       RCM06680
      TCNTCT=XCNTCT/ABS(UCNTCT)                                         RCM06690
      AHEAD=A(199)                                                      RCM06700
      THEAD=1.0/A(199)                                                  RCM06710
      IF(TCNTCT.LE.THEAD) GO TO 23                                      RCM06720
      RETURN                                                            RCM06730
   23 JCOUNT=JCOUNT+1                                                   RCM06740
      SWR=1                                                             RCM06750
      WRITE(6,24)                                                       RCM06760
      WRITE(6,25)N,TIME,SWL,SWR,JCOUNT                                  RCM06770
   24 FORMAT(5X,'H.P. GAS IN PORT CLOSES AT')                           RCM06780
   25 FORMAT(5X,I4,5X,F9.7,5X,3I3)                                      RCM06790
      RETURN                                                            RCM06800
   26 CONTINUE                                                          RCM06810
   30 IF(JCOUNT.EQ.4) GO TO 80                                          RCM06820
      IF(JCOUNT.EQ.6) GO TO 90                                          RCM06830
      IF(JCOUNT.EQ.8) GO TO 100                                         RCM06840
      IF((R(2)-R(4)).GT.0.1) GO TO 31                                   RCM06850
      RETURN                                                            RCM06860
   31 JCOUNT=JCOUNT+1                                                   RCM06870
      SWL=5                                                             RCM06880
      WRITE(6,32)                                                       RCM06890
      WRITE(6,33)N,TIME,SWL,SWR,JCOUNT                                  RCM06900
   32 FORMAT(5X,'HP AIR OUT PORT CLOSES AND TUNING PORT L1 OPENS AT')   RCM06910
   33 FORMAT(5X,I4,5X,F9.7,5X,3I3)                                      RCM06920
      RETURN                                                            RCM06930
   40 IF(JCOUNT.EQ.7) GO TO 110                                         RCM06940
      IF(U(3).GE.0.0) GO TO 41                                          RCM06950
      RETURN                                                            RCM06960
   41 JCOUNT=JCOUNT+1                                                   RCM06970
      PSEXIT=PSOUT2                                                     RCM06980
      SWL=2                                                             RCM06990
      WRITE(6,42)                                                       RCM07000
      WRITE(6,43)N,TIME,SWL,SWR,JCOUNT                                  RCM07010
   42 FORMAT(5X,'TUNING PORT L1 CLOSES AND EXHAUST PORT E1 OPENS AT')   RCM07020
   43 FORMAT(5X,I4,5X,F9.7,5X,3I3)                                      RCM07030
      RETURN                                                            RCM07040
```

```
   80 IF(ABS(U(201)).GT..0001) GO TO 81
      RETURN
   81 JCOUNT=JCOUNT+1
      SWR=5
      WRITE(6,82)
      WRITE(6,83)N,TIME,SWL,SWR,JCOUNT
   82 FORMAT(5X,'TUNING PORT R1 OPENS AT')
   83 FORMAT(5X,I4,5X,F9.7,5X,3I3)
      RETURN
   50 IF(JCOUNT.EQ.9) GO TO 120
      IF(ABS(U(201)-U(3)).LE.0.001) GO TO 51
      RETURN
   51 JCOUNT=JCOUNT+1
      SWR=1
      WRITE(6,52)
      WRITE(6,53)N,TIME,SWL,SWR,JCOUNT
   52 FORMAT(5X,'TUNING PORT R1 CLOSES AT')
   53 FORMAT(5X,I4,5X,F9.7,5X,3I3)
      RETURN
   90 THEAD1=X(201)/(ABS(U(3))+A(3))
      KCOUNT=KCOUNT+1
      IF(KCOUNT.EQ.1) TTOT1=TTOTAL
      IF(TTOTAL.GE.(TTOT1+THEAD1)) GO TO 91
      RETURN
   91 JCOUNT=JCOUNT+1
      SWL=5
      WRITE(6,92)
      WRITE(6,93)N,TIME,SWL,SWR,JCOUNT
   92 FORMAT(5X,'EXHAUST PORT E1 CLOSES AND TUNING PORT L2 OPENS AT')
   93 FORMAT(5X,I4,5X,F9.7,5X,3I3)
      RETURN
  110 IF(U(3).GE.0.0) GO TO 111
      RETURN
  111 JCOUNT=JCOUNT+1
      PSEXIT=PSOUT3
      SWL=2
      WRITE(6,112)
      WRITE(6,113)N,TIME,SWL,SWR,JCOUNT
  112 FORMAT(5X,'TUNING PORT L2 CLOSES AND EXHAUST PORT E2 OPENS AT')
  113 FORMAT(5X,I4,5X,F9.7,5X,3I3)
      RETURN
  100 IF(ABS(U(201)).GT..0001) GO TO 101
      RETURN
  101 JCOUNT=JCOUNT+1
      SWR=5
      WRITE(6,102)
      WRITE(6,103)N,TIME,SWL,SWR,JCOUNT
  102 FORMAT(5X,'TUNING PORT R2 OPENS AT')
  103 FORMAT(5X,I4,5X,F9.7,5X,3I3)
      RETURN
  120 IF(ABS(U(201)-U(3)).LE.0.0001) GO TO 121
      RETURN
  121 JCOUNT=JCOUNT+1
      SWR=4
```

```
      WRITE(6,122)                                                      RCM07590
      WRITE(6,123)N,TIME,SWL,SWR,JCOUNT                                 RCM07600
  122 FORMAT(5X,'TUNING PORT R2 CLOSES AND L.P. AIR INLET OPENS AT')    RCM07610
  123 FORMAT(5X,I4,5X,F9.7,5X,3I3)                                      RCM07620
      RETURN                                                            RCM07630
   60 IF((R(4)-R(2)).GT.0.1) GO TO 61                                   RCM07640
      RETURN                                                            RCM07650
   61 JCOUNT=JCOUNT+1                                                   RCM07660
      SWL=1                                                             RCM07670
      WRITE(6,62)                                                       RCM07680
      WRITE(6,63)N,TIME,SWL,SWR,JCOUNT                                  RCM07690
   62 FORMAT(5X,'EXHAUST PORT E2 CLOSES AT')                            RCM07700
   63 FORMAT(5X,I4,5X,F9.7,5X,3I3)                                      RCM07710
      RETURN                                                            RCM07720
   70 IF(U(201).GE.0.0) GO TO 71                                        RCM07730
      RETURN                                                            RCM07740
   71 JCOUNT=0                                                          RCM07750
      SWR=1                                                             RCM07760
      WRITE(6,72)                                                       RCM07770
      WRITE(6,73)N,TIME,SWL,SWR,JCOUNT                                  RCM07780
   72 FORMAT(5X,'CYCLE COMPLETED.')                                     RCM07790
   73 FORMAT(5X,I4,5X,F9.7,5X,3I3)                                      RCM07800
      RETURN                                                            RCM07810
      END                                                               RCM07820
```

APPENDIX B

PROGRAM RCM

B.1. Program Description

B.1.1. Computational Grid

The computational region is divided into 100 cells; the solution grid points are odd numbered, e.g., 3, 5, 7 ..., 201 with 1 and 203 being the points where the boundary conditions are specified. The even numbered points, 2, 4, 6 ..., 202 are intermediate locations where solutions are stored before being assigned to the solution grid points. See Fig. (2).

B.1.2. Data Input

Data for various ports (exhaust, inlet, etc.) is specified in dimensional form in S.I. units (Pascal (N/m^2) for pressure, kg/m^3 for density, m/s for velocity etc.). Reference values are also specified in like manner. See lines RCM00210 through 00250.

Initial data is specified through a call to an appropriate subroutine, depending on where the calculation is started for a particular wave diagram. For the example given in section II on the Spectra Technology wave diagram, the computation is started at the point when the high pressure gas inlet port just opens. The call for initial data is made to subroutine INIT3R, which prescribes data consistent with a solid wall boundary at the left and a 'piston' inflow boundary at the right.

B.1.3. Non-dimensionalization

Non-dimensionalization is carried out in lines 00540 through 00610 with entropy defined as

$$S = \ln \left(\frac{p}{\rho^{\gamma}}\right)$$

Note that velocities are all referred to a reference sonic velocity defined by

$$a_{ref} = \frac{p_{ref}}{\rho_{ref}}$$

B.1.4. Structure

The main program loop starts at line 00630, for the number of time steps specified. The time step is computed according to the appropriate CFL condition for the method, and a random number for the time step is generated by a call to the function subroutine WDP.

A secondary loop to define the sequence of local Riemann problems for the time step is set up at line 00750. For each Riemann problem defined, a call is made to subroutine GLIMM which i) solves the Riemann problem, and ii) samples the solution using the random number generated. The subroutine then returns the sampled solution as the parameters PGLIM, RGLIM, UGLIM for the pressure, density and velocity respectively. These solutions are initially stored in the even numbered intermediate locations on the grid, and are then assigned to either the left or the right solution grid point depending on whether the random number was in the negative or the positive half of the interval respectively.

A call is then made to one of the modular subroutines structured for particular types of wave diagrams, lines 01050-01080, and the others are commented out.

Boundary conditions are invoked after the call to the modular subroutines which return the proper values of the switches SWL and SWR. The structure of the boundary condition subroutines is described in section II. This sequence completes one pass through the main loop and the process is repeated for the number of time steps specified.

B.2. Example Use of Program RCM

The program is set up in the following steps:

i)	Line 00150 -	output device designation. See B.3.
ii)	Line 00190 -	specify the number of time steps, k , and the switches SWL and SWR consistent with where the computation is to be started.
iii)	Lines 00210 - through 00250	prescribe flow data for various ports in dimensional form. See list of variables for explanation of variable names.
iv)	Lines 00490 - through 00530	invoke the proper initial data subroutine and comment out the rest. See list of subroutines for explanation of subroutine, function subprogram names.
v)	Line 00660 -	set the interval for number of time steps at which a plot of the flow parameters is required.
vi)	Lines 01050 - through 01080	user supplied modular subroutine for particular wave diagram to be computed. Comment out the rest.
vii)	Line 01190 -	call to plotting routine should be consistent with interval specified in line 0660.
viii)	Lines 02650 - through 03700	identify proper subroutine to prescribe initial data (consistent with iv), and specify the data in the subroutine in dimensional form.
ix)	Lines 05470 - through 05990	specify plotting parameters, viz., origins of plots, scales, number of points to be plotted, color of plots, etc. Facility dependent. The subroutines PLOT1 and PLOT2 given in the listing are structured for DISSPLA software installed in the facility at NPGS.
x)	Lines 06040 - through 07820	user supplied modular subroutine for wave diagram to be computed.

B.3. Execution

The program is run in an interactive mode and is invoked through a call to DISSPLA, available on most mainframes. After compiling the program

(FORTRAN H Extended compiler), the following command executes it:

DISSPLA filename

If working at stations equipped with dual screens, the command on line 150 can be of the type

CALL TEK618 → Tektronix screen

If working on a non-graphics terminal, or a single screen station, this should be changed to

CALL COMPRS

which generates a 'DISSPLA METAFILE' to be routed later to either a screen or a plotter, e.g., VRSTEC, IBM79, TEK618, etc. Once generated, the metafile can be accessed and routed by the command

DISSPOP device designation

These are facility dependent commands and should be modified accordingly.

B.4. List of Important Variables (In Alphabetical Order)

A	-	sonic velocity
AHEAD	-	sonic speed of head wave of rarefaction fan
AMACH	-	Mach number
AL	-	left side sonic speed value for RP
AR	-	right side sonic speed value for RP
AREF	-	reference speed of sound
ASTAR	-	speed of sound in 'starred' state of RP solution (see Fig. 3)
CFLNUM	-	CFL number for time step determination
DT	-	time step
DX	-	grid cell width
EPS	-	small number for pressure iteration in RP solver
G	-	ratio of specific heats, γ
IDIGT	-	see WNORM
II	-	argument used in function subprogram PHI equal to either 0 or 1
JCOUNT	-	counter
K	-	number of time steps
KCOUNT	-	counter
N	-	counter for time steps
N1	-	counter

P - pressure
PA - flow parameter describing 'a' state in transition functions
PGLIM - pressure value returned by subroutine GLIMM
PL - left side pressure value for RP
PR - right side pressure value for RP
PREF - reference pressure
PSEXIT - static pressure at exit or outlet port
PSINL - static pressure for incoming 'piston' flow on left side
PSINR - static pressure for incoming 'piston' flow on right side
PSOUTn - n = 1,2,3 - exit static pressures for cycles with more than one exhaust port
PSTAR - pressure in 'starred' state of RP solution (see Fig. 3)
PTOTIN - total pressure for isentropic inflow
QPRINT - specification of interval size for output
QSTOP - maximum number of iterations for solution of Riemann problem, (RP)
R - density
RA,RB - flow parameters describing 'a' and 'b' states in transition functions
RGLIM - density returned by subroutine GLIMM
RINL - static density for incoming 'piston' flow on left side
RINR - static density for incoming 'piston' flow on right side
RL - left side density for RP
RMU - function of γ
RR - right side density for RP
RREF - reference density
RTOTIN - total density for isentropic inflow
S - entropy
SWL - switch for left boundary
SWR - switch for right boundary
TCNTCT - time taken by contact surface to travel a certain distance
THEAD - time taken by head wave of expansion to travel a certain distance
TIME - real time in seconds
TIMEREF - reference time
TTOTAL - cumulative non-dimensional time for number of time steps
U - velocity
UA - flow parameter for 'a' state in transition functions
UCNTCT - velocity of contact surface
UEXMAX - maximum velocity occurring at an outflow boundary
UGLIM - velocity returned by subroutine GLIMM
UL - left side velocity for RP
UR - right side velocity for RP
USTAR - velocity in 'starred' state of RP solution (see Fig. 3)
WDP - value returned by random number generator subprogram
WL - left shock wave velocity
WR - right shock wave velocity
WNORM - variable used in random number generator subprogram
X - space dimension
XCNTCT - location of contact surface
XI,XII - random numbers scaled to grid cell

XREF - reference length
Y - argument used in function subprogram PHI equal to PSTAR
Z - argument used in function subprogram PHI equal to either PL or PR
ZETA - dummy variable (for initialization purposes in random number generator)

B.5. List of Subroutines, Function Subprograms

B.5.1. Subroutines

INIT1 - prescribes initial data corresponding to SWL=1, SWR=1; e.g., shock-tube problem

INIT2L - prescribes initial data corresponding to SWL=2, SWR=1

INIT2R - prescribes initial data corresponding to SWL=1, SWR=2

INIT3L - prescribes initial data corresponding to SWL=3, SWR=1

INIT3R - prescribes initial data corresponding to SWL=1, SWR=3

PLOT1,2 - graphics subroutines

GLIMM - solves the Riemann problem, samples the solution and returns values for flow parameters

GE - modular user supplied subroutine to simulate wave diagram of General Electric Wave Engine

DETON - modular user supplied subroutine to simulate evacuation of detonation chamber

SPCTRA - modular user supplied subroutine to simulate wave diagram of Spectra Technology's Pressure Exchanger

BCL1 - prescribes boundary conditions (BC's) corresponding to SWL=1, i.e., solid wall on left side

BCL2 - prescribes BC's corresponding to SWL=2, i.e., outflow at constant static pressure on left side

BCL3 - prescribes BC's corresponding to SWL=3, i.e., 'piston' inflow on left side

BCL4 - prescribes BC's corresponding to SWL=4, i.e., isentropic inflow from reservoir on left side

BCL5 - prescribes BC's corresponding to SWL=5, i.e., wave 'tuning' on left side

BCR1, BCR2, BCR3, BCR4, BCR5 - prescribe BC's corresponding to SWR=1,2,3,4,5 respectively on right side

B.5.2. Function Subprograms

PHI(y,z) - required in iteration procedure for solution of RP

PHI1(PB) - describes shock transition function, $\varphi_a(p_b)$, for two states a and b connected by a shock wave (see Ref. 6, Ch. III)

PSI(PB) - describes rarefaction transition function, $\psi_a(p_b)$, for two states a and b connected by a rarefaction wave (see Ref. 6, Ch. III)

WDP(II) - generates a random number in a van der Corput sequence each time it is invoked. Note that it needs to be called once from outside the main loop by specifying an argument II=1 to initialize IDIGT and WNORM, returning a value of 0 for the dummy variable ZETA, and then a second time from within the main loop with an argument II=0 to return a value which is the random number.

END
FILMED

5-86

DTIC